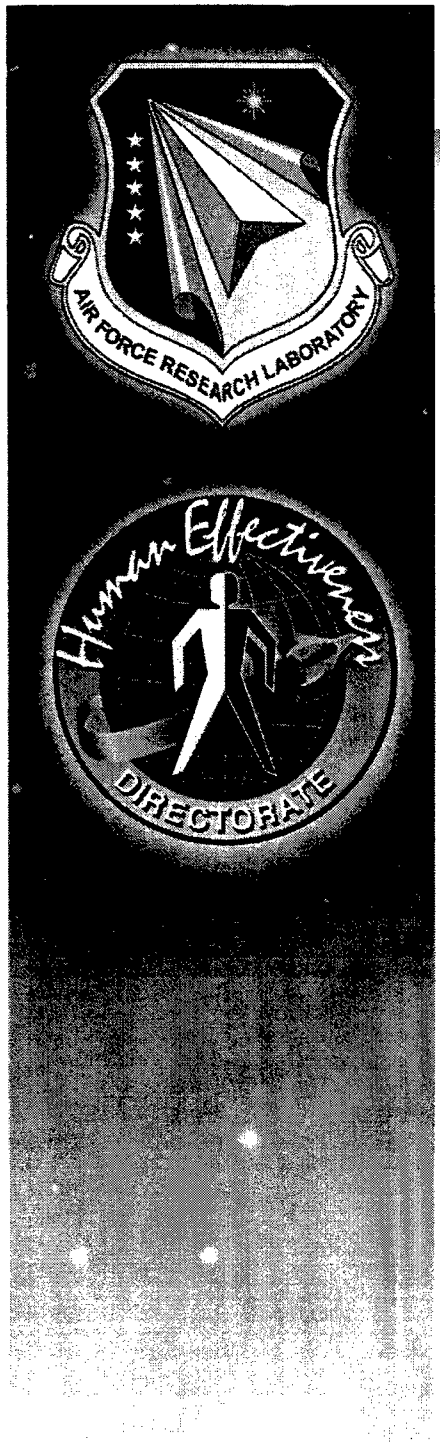
AFRL-HE-WP-TR-2005-0120

AIR FORCE RESEARCH LABORATORY



Inheriting Constraint in Hybrid Cognitive Architectures: Applying the EASE Architecture to Performance and Learning in a Simplified Air-Traffic Control Task

Ronald S. Chong

Department of Psychology, MS 3F5
George Mason University
4400 University Drive
Fairfax, VA 22030-4444

February 2004

Final Report for October 2001 to February 2004

Human Effectiveness Directorate
Warfighter Interface Division
Cognitive Systems Branch
2698 G Street
Wright-Patterson AFB OH 45433-7604

NOTICES

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data, does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them

This report was cleared for public release by the Air Force Research Laboratory Wright Site Public Affairs Office (AFRL/WS) and is releasable to the National Technical Information Service (NTIS). It will be available to the general public, including foreign nationals.

Please do not request copies of this report from the Air Force Research Laboratory. Additional copies may be purchased from:

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161

Federal Government agencies and their contractors registered with the Defense Technical Information Center should direct requests for copies of this report to:

Defense Technical Information Center
8725 John J. Kingman Road, Suite 0944
Ft. Belvoir, VA 22060-6218

TECHNICAL REVIEW AND APPROVAL

AFRL-HE-WP-TR-2005-0120

This technical report has been reviewed and is approved for publication.

FOR THE DIRECTOR

//SIGNED//

BRADFORD P. KENNEY, Lt Col, USAF
Deputy Chief, Warfighter Interface Division
Human Effectiveness Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
February 2004	Final	October 2001 - February 2004

4. TITLE AND SUBTITLE
Inheriting Constraint in Hybrid Cognitive Architectures: Applying the EASE Architecture to Performance and Learning in a Simplified Air-Traffic Control Task

5a. CONTRACT NUMBER
F33615-01-C-6077

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62202F

6. AUTHOR(S)
Ronald S. Chong

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
1710D111

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Department of Psychology, MS 3F5
George Mason University
4400 University Drive
Fairfax, VA 22030-4444

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory, Human Effectiveness Directorate
Warfighter Interface Division
Air Force Materiel Command
Cognitive Systems Branch
Wright-Patterson AFB OH 45433-7604

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-HE-WP-TR-2005-0120

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report describes the development and evaluation of models of behavior in the AMBR air-traffic control (ATC) and category learning task. We adopt, as constraints, several of Newell's principles and recommendations on the development and use of models and architectures. The EASE architecture, developed in the course of the AMBR project, came about as a result of the constraints of "listening to the architecture" and identifying areas where the set of mechanisms of the original architecture (Soar) needed to be amended. This work also reuses existing Soar models of category learning (SCA), and operationalizes and expands a non-architectural model found in the literature (RULEX).

15. SUBJECT TERMS AMBR, Cognitive Modeling, Concept Acquisition, Category Learning, Cognitive Architecture, Cognitive Modeling, Multiple-Task Performance, SCA, RULEX, EPIC, EASE, ACT-R

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			John L. Camp
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR	52	19b. TELEPHONE NUMBER *(include area code)* (937)255-7773

THIS PAGE LEFT INTENTIONALLY BLANK

TABLE OF CONTENTS

List of Figures

List of Tables

THIS PAGE LEFT INTENTIONALLY BLANK

Inheriting Constraint in Hybrid Cognitive Architectures

Applying the EASE Architecture to Performance and Learning in a Simplified Air-Traffic Control Task

1.0 INTRODUCTION

This report describes the development and evaluation of models of behavior in the AMBR air-traffic control (ATC) and category learning task. The emphasis of the report will be on modeling constraints derived from our methodology. Our overall modeling philosophy is driven by cognitive architectures as theories of human perception, cognition, and action. Architectures are critical to the development of broad, comprehensive theories Allen Newell called *unified theories of cognition*, or UTCs (Newell, 1990).

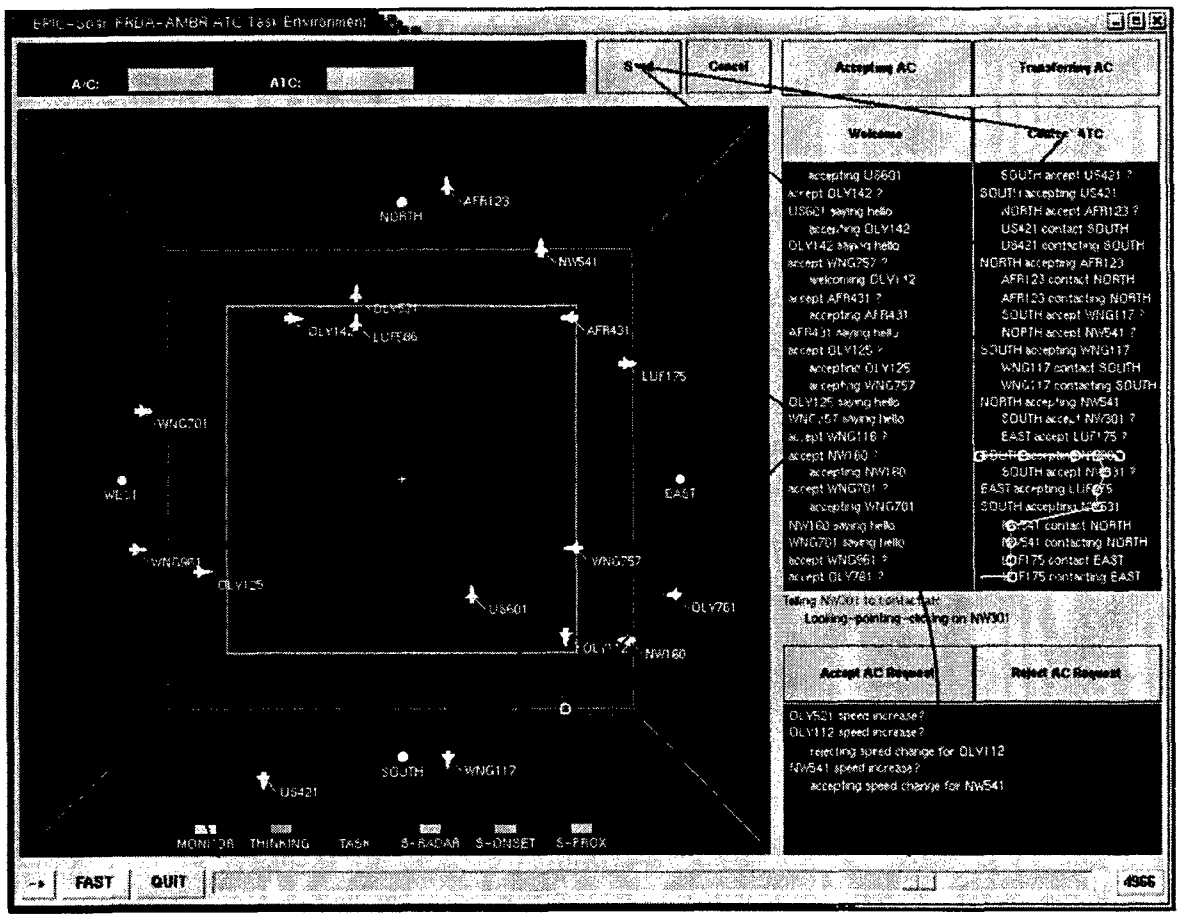


Figure 1: The enroute air traffic control task environment. Our EPIC-Soar eyeball and mouse pointer are overlaid on the environment.

Cognitive architectures, as instantiations of UTCs, comprise a set of fixed (or slowly evolving) mechanisms and representations on which models of a wide range of behavior can be built.

We adopt, as constraints, several of Newell's principles and recommendations on the development and use of models and architectures. First is "listening to the architecture" or making a commitment to an architecture's mechanisms. When modeling a new behavior or phenomena, one must use the existing mechanisms rather than introduce new mechanisms solely to address the requirements of the model or to fit data. However, only when behavior cannot be plausibly implemented using an architecture's existing mechanisms should the set of mechanisms be amended, either by modifying existing mechanisms or adding new mechanisms. In this work, when considering architectural change, we followed an integrative approach, incorporating validated components from other architectures rather than modifying the architecture less conservatively. EASE, the architecture developed and used here, combines elements of ACT-R, Soar, and EPIC into one integrated hybrid architecture.

For architectures to have theoretical power, results and validations from previous model implementations must cumulate as constraints on future modeling efforts (Newell 1990). This principle of cumulation as constraint led us to reuse existing models. In this work, the ATC performance model from AMBR Phase I was reused. For the category learning component, one previously developed model was reused and another, not originally developed in an architectural theory, adapted to EASE. The principle of cumulation also applies to architectural mechanisms: EASE brings together theoretical strengths of several existing architectures, incorporating both the mechanisms and common parameter settings in an explicit attempt to inherit the validation and consequent constraint of multiple architectures. Cumulation reduces flexibility in creating models, but increases the predictive and theoretical power of architectural models.

Taking architectural constraint seriously during model creation and refinement leads to an emphasis of explanation over fitting. Freely changing architectural mechanisms, model knowledge, and model and architecture parameters may lead to better fits to the data, but not necessarily to an improved understanding of the underlying phenomena. We deliberately chose to minimize such changes, fixing model knowledge, parameter settings, and the architecture to the extent possible. The positive consequence of such constraint is clearly evident by the use of the Symbolic Concept Acquisition (SCA) model. Although the initial prediction of the aggregate human data was quite poor, we resisted abandoning the model or radically reformulating it or the architecture, choosing instead to perform a fine-grained analysis of the individual human data. This analysis revealed that the model did match the learning trajectories of some individual subjects and that some subjects were considering factors that the task instructions directed them to ignore. Thus, the architectural constraint led to significantly broader understanding of the human behavior in the task.

In summary, the focus of this work was to understand architectural and task constraints and to develop plausible models that took these constraints into account.

2.0 SYSTEM-LEVEL ANALYSIS OF THE TASK

Like most complex tasks humans perform, the combination of the ATC and category learning tasks rely on multiple human systems—visual sensation and perception, memory, cognition, and action—and the interaction of such systems (e.g. eye-hand coordination).

Our first model development step was to assess the influence of each of these systems on task performance. This preliminary assessment is useful because it:

- informs the selection of a modeling framework or architecture;
- provides the modeler with a sense of the model's eventual complexity;
- helps identify existing empirical task-relevant behavioral data that can be modeled;
- points to existing models that might be reused and further validated.

This evaluation step, based on a functional analysis of the task and empirical studies that show connections between systems and behavior, provides qualitative bounding constraints for the modeling effort.

The ATC task has a strong visual perceptual component.[1] In functional terms, the eyes are responsible for finding features in the world that will trigger task-relevant behavior. Eye scan patterns affect what can be seen and when it can be seen. Therefore, we hypothesized that perception must have a significant influence over task performance.

The memory system also plays a key role in this task. There is often a long delay between attending to a blip on the display and performing an action on it. For example, when handing off an aircraft, one of the important features to be remembered is what action was last performed on the aircraft. Another feature is blip location. The volatility of human memory in situations such as these is readily observed in subjects and reported by them. Memory effects influence the overall task performance.

Knowledge in the cognitive system is inherently related to performance because it provides the strategies and decision-making processes for the task. In addition, the cognitive system can have strategies for *coping* with the limitations of the perceptual and memory systems. For example, memory rehearsal can be used to enhance one's ability to recall an item. Although all subjects should have the same task knowledge (per the task instructions), they can also employ vastly different knowledge (e.g. pre-existing heuristics; their own

[1] Throughout this report, we use the term *perception* to refer to the sensory, perceptual, *and* the ocular motor functional of the eye.

resolution of ambiguities in the task) and biases (driven by motivation, personality, etc.) Arguably the greatest source of within and between-subject variability is due to by the knowledge.

The contribution of the manual motor system (hand movements) to performance was expected to be insignificant for the ATC tasks. We came to this conclusion because once a task action sequence is triggered, its constituent steps can (nominally) be performed ballistically, without intervening reasoning, producing roughly the same execution times regardless of the task conditions.[2] Therefore, we could have represented motor behavior as a simple, constant-time process. However, because mouse movements influence eye-scan patterns through the eye-hand coordination required to move the mouse, we modeled manual motor behavior.

Other human "systems" could be considered in this kind of analysis. For example, the influence of the motivational, emotional, physiological (e.g. fatigue, stress, etc.) "systems" could have been assessed. We instead made the simplifying assumption that, on average, these systems had an insignificant effect on overall performance.

Taken together, this qualitative analysis suggested a significant influence of the perceptual, memory, cognitive, and motor systems on task performance, not only due to the individual systems but also the interaction of those systems. From this we conclude that we should construct a model that not only captures the details of the individual systems but can also their interaction.

3.0 ARCHITECTURE

The preceding analysis indicated we should use an architecture that provided psychologically-plausible implementations of perception, cognition, memory, and motor systems. EPIC-Soar (Chong & Laird, 1997), a combination of EPIC (Kieras & Meyer, 1994; Meyer & Kieras, 1997a, 1997b) and Soar (Rosenbloom, Laird, & Newell, 1993; Newell, 1990), provided all of these elements except for a straightforward account of memory effects (forgetting).

During the course of the AMBR project, EPIC-Soar was extended to include base-level learning, the fundamental memory mechanism in ACT-R (Anderson & Lebiere, 1998), to provide a validated account of memory volatility and retention. With this addition, EPIC-Soar was renamed to EASE, for Elements of ACT-R, Soar, and EPIC. EASE is a hybrid system, incorporating both symbolic and subsymbolic representations and mechanisms. We briefly describe each of these architectures, in order of their integration into EASE, concentrating only on aspects relevant to AMBR.

[2] When task action execution time was later analyzed, no significant effect of task condition (aided/unaided) or difficulty was found, confirming this assumption.

3.1 Soar

Soar is a general architecture for building artificially intelligent systems and for modeling human cognition and behavior (Rosenbloom, Laird, & Newell, 1993; Newell, 1990). Soar has been used to model central human capabilities such as learning, problem solving, planning, search, natural language, and HCI tasks.

Soar is a production system that has been significantly extended to include mechanisms and structures believed to be functionally necessary for producing intelligent behavior. The processing cycle consists of a search for operators, the selection of a best operator, and the application of the operator. Operators encode persistent actions in Soar, and generally correspond in function to the productions of ACT-R and EPIC. Soar has two memory representations: procedural memory is represented by production rules, and declarative memory represented by attribute-value structures.

There are occasions when knowledge search is insufficient and does not lead to the selection or application of an operator. This situation is called an *impasse.* To resolve an impasse, Soar automatically creates a subgoal where processing focuses on selecting and apply operators to resolve the impasse in order that processing in the parent goal can resume.

Soar incorporates a single learning mechanism called *chunking.* Chunking compiles the results of problem solving in a subgoal into new production rules. When combined with various problem solving methods, chunking has been found to be sufficient for a wide variety of learning (Wray & Chong, 2003; Chong, 1998; Miller & Laird, 1996; Lewis, *et al.*, 1990).

One weakness of Soar is the difficulty of producing memory effects such as forgetting. In humans, forgetting is the default, non-deliberate condition, while remembering requires an effortful process (e.g. rehearsals or the use of a reliable encoding) or multiple exposures of the stimuli. Soar's declarative memory system has exactly the opposite properties: remembering is the default condition while forgetting requires the deliberate act of removing items from memory. Although Soar's present memory mechanisms can account for some memory effects (Young & Lewis, 1999), the architecture does not require models to follow such accounts. Performance in the ATC task, based on the system-level analysis, appears to be strongly influenced by memory effects. Therefore, we chose to explore an architectural change to EPIC-Soar for producing such memory effects.

3.2 EPIC

In contrast to Soar, which is theoretically silent on the topics of perception and action, EPIC's perceptual and motor processes provide sophisticated accounts of the capabilities and constraints of these systems. EPIC (Executive Process-Interactive Control) (Kieras & Meyer, 1997; Meyer & Kieras, 1997a, 1997b) is an architecture whose primary goal is to account for detailed human dual-task performance. It extends the work begun with the Model Human Processor, MHP (Card, Moran, & Newell, 1983). Like MHP, EPIC

consists of a collection of processors and memories. There are three classes of processors: perceptual, cognitive, and motor. However, the EPIC processors and memories are much more elaborate, each representing a synthesis of empirical evidence and theories. Unlike MHP, EPIC, being an architecture, can be programmed and executed.

EPIC includes three perceptual processors—visual, auditory, and tactile. These receive input from simulated physical sensors. The visual perceptual processor, which is of particular interest for the ATC task, represents the eye's retinal zones (bouquet, fovea, parafovea, periphery) and the constraint of feature availability as a function of retinal zone.

The output of perceptual processors is sent to the cognitive processor. The cognitive processor consists of working memory, long-term memory, production memory, and a multi-match, multi-fire production system. The cognitive processor performs task reasoning and initiates actions by sending output commands to the motor processors: ocular, vocal, and manual.

Similar to Soar, EPIC uses a memory system where persistence is the default. The cognitive processor has no learning mechanism. The merging of EPIC and Soar provided a system that gave good coverage of perception, cognitive, learning, and motor behavior. The missing component, for the AMBR task, was a plausible memory system

3.3 ACT-R

ACT-R (Anderson & Lebiere, 1998) is a hybrid architecture that implements a theory of cognitive adaptation. ACT-R, at the symbolic level, like Soar and EPIC, is a production system, representing procedural knowledge as production rules and declarative knowledge as attribute-value memory structures. ACT-R features many subsymbolic mechanisms that each address a specific form of cognitive adaptation. In general, these mechanisms modulate the availability of symbolic elements (declarative and procedural), as well as the time to retrieve these elements from memory.

One of these mechanisms is called *base-level learning*. It assigns each declarative memory element an *activation*. The activation learning mechanism varies the activation of each chunk as a function of its recency and frequency of use. When a memory element is created, it is assigned an initial level of activation. The activation begins to decay exponentially as a function of time. If the activation falls below the *retrieval threshold*, the memory element cannot be retrieved and is effectively forgotten. As a consequence, the memory element is not available to satisfy the conditions of a production rule. A memory element's activation is boosted through several avenues: use (through task-related recall); spreading activation from associated memory elements; activation noise. The decay process immediately resumes after an element's activation has been boosted.

3.4 EASE (Elements of ACT-R, Soar, and EPIC)

ACT-R, Soar and EPIC provide unique strengths that EASE combines into one hybrid, integrated architecture. EPIC has perceptual and motor processors but, presently a non-learning cognitive processor. Soar, in contrast, has no perceptual or motor processors but is a learning cognitive architecture. ACT-R contains mechanisms for representing memory effects that exist in neither EPIC nor Soar. EASE (Figure 2) is an integration of the sensory, perceptual and motor processors of EPIC, one of the memory mechanisms of ACT-R, and the cognitive mechanisms of Soar.

In EASE, cognition (Soar) receives perceptual and motor processor messages (from EPIC) as input to its working memory, and returns motor processor commands to the motor processors (EPIC) as output based on the processing of the inputs. EASE's version of ACT-R's base-level learning mechanism is controlled by this activation equation:

Equation 1: $$A_i = \beta + \ln(\sum_j t_j^{-d}) + \varepsilon$$

The primary difference between Equation 1 and the ACT-R activation and base-level learning equations found in Anderson and Lebiere (1998) is that the spreading activation component is not used. (ACT-R's

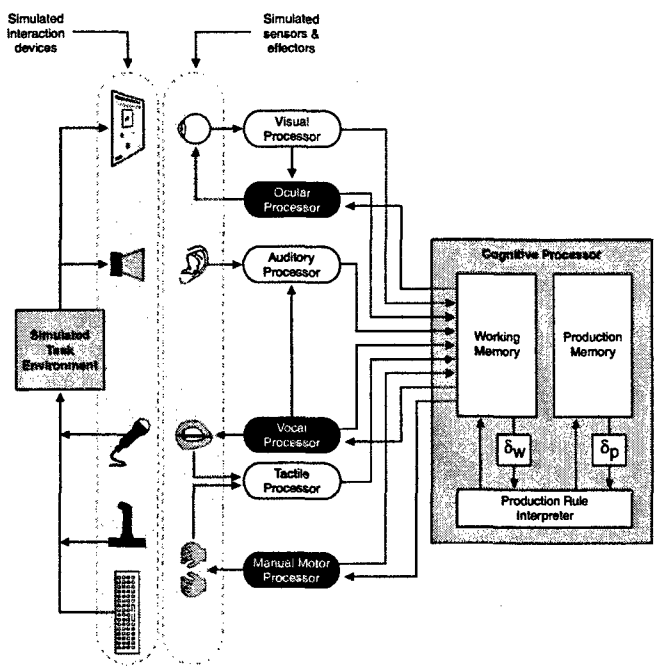


Figure 2: Diagram of the EASE hybrid architecture. The sensory, perceptual and motor processors are provided by EPIC; Soar provides the cognitive control and symbolic learning; the base-level learning mechanism of ACT-R provides subsymbolic modulation of declarative and procedural knowledge through the base-level learning mechanism, represented as δp and δw.

spreading activation and association mechanisms have not yet been incorporated into EASE.) The base-level learning mechanism is controlled by four free parameters, three of which are variables in the activation equation:

- *Base-level constant* (β): this value specifies the initial activation given to a newly created memory element. β in our model was set to 1.0, a value used in many ACT-R models.
- *Learning rate* (d): the rate of activation decay. The ACT-R default of 0.5 was used.
- *Transient noise* (ε): noise is sampled from a zero-centered logistic distribution and added to an element's activation. A commonly used ACT-R value of 0.25 was used.
- *Retrieval threshold*: when activation falls below this value, the memory element cannot be retrieved and is effectively forgotten. The ACT-R default value of 0.0 was used.

Importantly, these parameter values were not tuned to produce the fits presented later in this report. In accordance with our modeling philosophy, these ACT-R values, determined through successful modeling of a wide range of behavior, were used as a constraint on the model building process.

EASE inherits the detailed predictions and theory embodied in the sensory, perceptual and motor systems from EPIC, the cognitive problem solving, planning, and symbolic learning capabilities of Soar, and the constraints of human memory provided the ACT-R mechanisms, as well as the reuse of default and commonly used free parameter values.

4.0 MODELING CATEGORY LEARNING: INTRODUCTION

The task environment used in AMBR Phase I was extended to include a category learning task. The learning task is isomorphic to the study performed by Nosofsky, Gluck, Palmeri, McKinley & Glauthier (1994). Figure 3 shows a comparison of their data to the AMBR learning task. A main effect of problem type was found in both Nosofsky, et al. (1994) and the AMBR data. However, the AMBR aggregate learning rates are much slower.

Several task differences might account for the dissimilar learning performance results. The stimuli in the original study were composed of three orthogonal features: shape (triangle or circle), size (small or large), interior (solid or dotted). In contrast, the instances used in the AMBR learning task consisted of contextually meaningful features, allowing subjects to benefit, or suffer, from using domain knowledge; e.g. "small planes (SIZE = S) should avoid high turbulence (TURB = 3) so their requests should be allowed (CATEGORY = ALLOW)." In addition to feature contextuality, the features also are more similar, relative to one another.

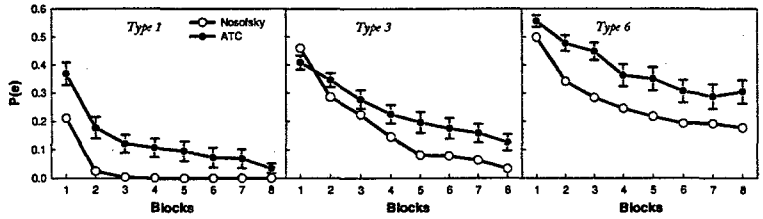


Figure 3: A comparison of the learning trajectories for Nosofsky, et al. (1994) and the ATC study.

All features were represented with alpha-numeric characters, and some features (e.g., S and 3, L and 1) share similar shapes.

A second task difference is the presence of a secondary task. Where subjects in Nosofsky, et al. (1994) performed only a category learning task, the AMBR subjects performed both a category learning and performance task. This may account for the learning difference; e.g., perhaps the secondary task "consumes" limited cognitive "resources", slowing learning. (This task was specifically designed to produce this kind of slowing.) However, AMBR control subjects—those exposed to the full learning and performance scenarios but who only performed the learning trials and ignored the ATC task—learned as slowly as non-controls. This finding suggests that reduced learning performance of the ATC task may not be due to the existence of the secondary ATC task.

A third task difference that may contribute to the performance difference is that subjects in Nosofsky, et al. (1994) were self-paced: a stimulus appeared, the subject made a classification, feedback was given, and the process repeated. Though not reported, an inter-stimulus time on the order of five seconds is a reasonable guess. In the AMBR learning task, sixteen altitude requests (category learning stimuli) were presented over a ten-minute scenario, giving an average inter-stimulus time of thirty-seven seconds; a seven-fold increase. Perhaps this differences in inter-stimulus time also contributes to the difference in learning rates.

4.1 Motivating an architectural approach for modeling category learning

There are many existing models that fit the Nosofsky, et al. (1994) or Shepard, Hovland, & Jenkins, (1961) data: Nosofsky, et al. (1994) present the fits of four exemplar-based models; Nosofsky, Palmeri & McKinley (1994) present the fit of RULEX, a hypothesis-testing model; Love & Medin (1998) report the fit of SUSTAIN, a network model of human category learning.

Any of these models, by virtue of demonstrating good quantitative fits to the Nosofsky, et al. (1994) data (and a number of other data sets), could have been recruited to the AMBR learning task. However, these models are all stand-alone models; they are purpose-built systems that perform only category learning and are not situated within a larger modeling framework. We prefer implementing models within the larger

theoretical context of a cognitive architecture. When used appropriately, the architectural approach addresses some of the limitations of many stand-alone models:

- *No account of process:* Few of these models are process models. Process models are desirable because they declare the individual steps and mechanisms—perceptual, cognitive, and motor steps—that define behavior. Most cognitive architectures, being based on production systems, naturally accommodate process models of behavior.
- *Unable to make time predictions:* Human data collected for the ATC task included the time to respond to an altitude change request; in other words, the time required to produce a category prediction. Response time predictions cannot be produced, in a principled way, by most stand-alone category learning models. However, production system based architectures have a cycle-based means of accounting for time. Time predictions are a by-product of model execution and a function of model complexity. If two models produce similar behavior (e.g., producing a category prediction) but one requires 100 cycles while the other requires 50 cycles, then their time predictions will be different. Therefore, time predictions serve as a critical post-hoc constraint in architecture-based models.
- *Human memory limitations ignored:* Few stand-alone learning models represent memory effects, such as forgetting. In some cases, memory effects have been simulated by imposing arbitrary constructs such as "capacity limits" or "probability of storage". Architectures may include primitive mechanisms that influence properties of memory. For example, the base-level learning mechanism of EASE (inherited from ACT-R) modulates the availability of knowledge. Therefore, memory effects such as apparent capacity limits, apparent probability of storage, forgetting, and priming can emerge from primitive memory mechanisms.
- *Large number of free parameters:* Models often rely on free parameters to improve their fit to data. Free parameters are placeholders for details that are yet to be uncovered or implemented. All things being equal, one would prefer a model with the fewest free parameters. Stand-alone models often contain a large number of free parameters. For example, RULEX contains ten free parameters that manipulate selection of a learning strategy, memory characteristics, and response error rates, among others. In contrast, the philosophical and theoretical pressures of architectural approaches encourage the parsimonious use of free parameters. Building models on a slowly evolving set of primitive mechanisms implies using a slowly evolving set of free parameters. Architectural mechanisms and their parameters are task-independent and therefore must apply to all models regardless of the behavior under study. Finally, as architecture-based models are validated, an acceptable range of values for each free parameter is often identified, providing further constraints.

- *Insensitivity to inter-stimulus time:* Stand-alone models are usually insensitive to time. They would predict the same performance if learning trials occurred ever 10 seconds or once per hour. The base-level learning mechanism in EASE is responsive to the recency and frequency of use of memories. Therefore, the availability of memories is affected by the timing of events in the environment or by the use of memories within a model.
- *Isolated learning:* It is unclear how stand-alone models could be made to be sensitive to the presence and possible interference of a competing task. This makes them inappropriate for modeling or predicting the effect a secondary task may have on learning. Architectures allow individual models to be executed together. EASE models of category learning were first developed in isolation, then integrated with the ATC model developed for Experiment 1. The architecture thus provides an environment for composing behavior and exploring if and how multiple tasks interact. Although architectures do not guarantee that combined models will produce and explain all psychologically-meaningful interactions, it does at least provide a framework and some constraints for this kind of exploration.

The remainder of this report presents the development of two architecture-based category learning models along with their fits and predictions. The first model is based on the reuse and extension of an existing Soar category learning model, Symbolic Concept Acquisition, SCA (Miller, 1993; Miller & Laird, 1996). The second is a new process model, RULEX-EM, inspired by RULEX, and incorporates both rule and exemplar representations as well as memory effects.

These models explore two alternate explanations for the learning differences depicted in Figure 3: the role of contextually meaningful features and the influence of inter-stimulus time. The SCA model posits that performance differences derive from the use of additional but irrelevant information available in the AMBR task; this additional information decreases the learning rate in the model. The RULEX-EM model explores the interaction of memory effects and prediction strategies to produce learning rates sensitive to problem difficulty. Both models emphasize the reuse and extension of existing category learning models, which is a critical aspect of the UTC philosophy. The key point of difference between the models is the role that knowledge and architectural mechanisms play in producing behavior, resulting in contrasting explanations of the data.

5.0 MODEL I: SYMBOLIC CONCEPT ACQUISITION (SCA)

Working within an architectural theory requires model reuse and cumulation (Newell 1990). Symbolic Concept Acquisition (SCA) is an existing model of category learning in the Soar theory. We adopted SCA because the architectural philosophy dictates that we should seek to reuse existing models and SCA is the

```
1.  instance = features and values /* from perception */
2.  while (no matching prediction rule for instance)
3.     abstract feature from instance
4.     remember most recently abstracted feature
5.  if (no feedback) return prediction else
6.     restore most recently abstracted feature to instance
7.     store new prediction rule for instance
```

Figure 4: A pseudocode representation of the SCA algorithm.

only extant Soar category learning model. Although developed over a decade ago, we were able to reuse the model's code.

A second important component of the modeling philosophy is to work within the constraints of the architecture and without introducing extra-architectural or new mechanisms. For example, because chunking is the Soar architecture's sole learning mechanism, we chose to limit ourselves to this mechanism alone for the SCA model. For this reason, the base-level learning mechanism in EASE was *not* used in the SCA model.

The original SCA model also included a production-based algorithm for simulating frequency effects. Because frequency effects are not an architectural component, we excised them from the model. Thus, an open question in Experiment 2 was to determine if this existing model could produce results that quantitatively matched human learning within the constraints of the architecture, and thus without introducing extra-architectural (or new) mechanisms. We removed the frequency effects but made no other changes to the model that would change the results reported by Miller & Laird (1996).

5.1 Description of SCA

Figure 4 presents a high-level representation of the SCA prediction and learning algorithm. The main body of the algorithm (lines 2-4) consists of a search for a matching prediction rule. The same search loop is used for both prediction and learning. Prediction is performed when feedback is not available; when category feedback is available, the model will refine its concept representation via learning. In contrast to the RULEX-EM model presented below, SCA does not use an explicit category representation. Instead, it focuses on the recall of prediction rules. Because all rules in Soar, including the prediction rules, are impenetrable, SCA models cannot report a category representation without additional introspection.

For prediction, SCA performs a specific-to-general search over previously learned prediction rules. As learning progresses, SCA learns more specific rules; i.e., rules that test more features. Thus, there may be rules at different levels of specificity. The algorithm first attempts to recall a prediction rule for all features (and feature values) in the instance (the condition in line 2). If no matching rule is retrieved, SCA enters the search loop. The first step is to abstract (ignore) a feature in the instance representation (line 3).

Available prediction rules:
1. (null) ➡ accept
2. (null) ➡ reject
3. fuel 20 ➡ accept

New instance:
size S, turbulence 3, fuel 20
category accept
Abstraction order: size, turbulence, fuel

New Prediction Rule:
4. fuel 20, turbulence 3 ➡ accept

Figure 5: Example of an SCA learning trial.

Abstracting the feature enables the search for less specific prediction rules. Determining which feature to abstract can occur in a number of different ways; the next section presents some of these options. The retrieve-abstract loop repeats until a matching rule is found. SCA includes rules to guess randomly when all features have been abstracted, so a matching rule will always be found.

When learning, SCA searches for a matching prediction rule as above. When SCA retrieves a rule matching the current instance, the prediction rule is specialized (6) by adding the last feature abstracted from the instance (remembered at 4). SCA stores this specialized rule as a new prediction rule (7). Over multiple learning trials, learning results in a general-to-specific search over the feature space. That is, SCA generally learns rules sensitive to one feature, then to two, and so on. The concept representation becomes more specific as more features (and combinations of features) are incorporated into learned prediction rules.

Figure 5 presents an SCA learning example. The example assumes that the model has previously learned a prediction rule (Rule 3) that indicates an instance with a fuel percentage of 20 should be accepted. A new positive instance (S,3,20) is presented. For this example, assume the abstraction order is size, then turbulence, then fuel. Because there are no matching prediction rules for all three features of the input instance, SCA abstracts size from the instance, leaving (3,20). Again, it looks for prediction rules for these features, and, finding none, abstracts the turbulence value, leaving (20). Rule 3 matches this instance. SCA now specializes Rule 3, adding the last abstracted feature and value (turbulence 3). The new rule, Rule 4, indicates that (3, 20) instances should be accepted. Had the example instance been negative, SCA would have learned a prediction rule that indicated instances with fuel values of 20 should be denied. In this case, given the previously learned prediction rule (i.e., FUEL=20 -> ACCEPT), the model would have come to recognize that fuel values of 20 could not be used, by themselves, to make category predictions. This situation does not reflect a contradiction, but rather that this feature alone cannot be used to make correct predictions.

Random Abstraction Order	Systematic Abstraction Order
0. fuel 20 ➡ accept	0. turb 1 ➡ accept
1. size L ➡ reject	1. turb 3 ➡ reject
2. fuel 40 ➡ reject	2. turb 1, fuel 20 ➡ accept
3. turb 1 ➡ accept	3. turb 1 ➡ reject
4. size S ➡ accept	4. turb 1, fuel 40 ➡ reject
5. size L, turb 1 ➡ reject	5. turb 3 ➡ accept
6. turb 1 ➡ reject	6. turb 3, fuel 20 ➡ accept
7. size L ➡ accept	7. turb 1, fuel 40, size S ➡ reject
8. fuel 20, turb 1 ➡ accept	8. turb 3, fuel 40 ➡ reject
9. size S ➡ reject	9. turb 1, fuel 20, size S ➡ accept

Figure 6: Example progression of rule learning in random (left) and systematic (right) abstraction orderings.

In summary, SCA is an incremental learner that creates rules in a general to specific manner with respect to instance features and values. As it is presented more examples, it acquires additional rules, gradually (but not monotonically) improving its category prediction performance. With enough training, SCA will eventually learn a maximally-specific rule (one that matches all the features of an exemplar) for each training instance. At this point, learning effectively ceases and SCA can readily predict the category of each exemplar by the use of its specific prediction rule. This "saturated" state represents what happens to subjects who are overtrained in a concept learning task; they memorize each training instance and its category.

5.2 Initial "out of the box" SCA model

We began applying SCA—without the simulated frequency effects but otherwise as described by Miller and Laird (1996)—to determine how well this minimalist model represented the human data. We concentrated only on the Nosofsky, et al. (1994) data initially and made no attempt to fit any ATC learning data in the first experiment. The goal was to assess the efficacy of the SCA model in capturing the ATC learning results without any "re-engineering" of the previous model for the new domain. Such reuse is necessary for the cumulation of results within an architectural theory.

The simulated frequency effects in the original SCA model were used to determine the abstraction order. Thus, having removed this aspect of the original model, we had to determine what abstraction order to use for the model. There were two obvious possibilities: random abstraction and systematic abstraction. With random abstraction order, the search over prediction rules is similar to a breadth-first search, generating many one-feature rules, then two-feature rules, etc. With a systematic abstraction order, the search is more like a depth-first search over prediction rules, specializing rules with the most relevant (last abstracted) feature, to ones with the two most relevant features, etc.

Examples of the progression of rule learning for the two types of abstraction orderings are shown in Figure 6. The random approach is generally slower than the systematic approach because the depth of the search space over all prediction rules is relatively shallow, relative to its breadth, and all leaf nodes represent

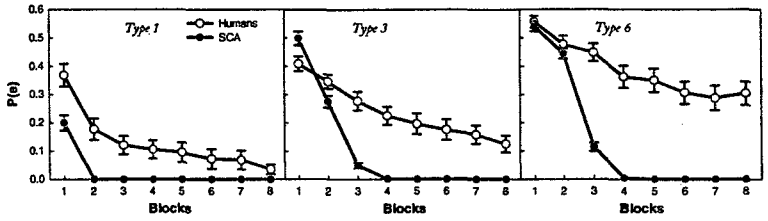


Figure 7: Initial SCA model results for the ATC learning task (G^2=673.62).

acceptable predictions (i.e., the experimental design assumed deterministic exemplars over all possible feature vectors). However, either resulted in significantly faster learning than found in Nosofsky, et al. (1994) except for Type I, in which the random feature abstraction was slower than the human learning.

One missing element in the original SCA model was any kind of hypothesis testing or relevant feature recognition. This omission was especially problematic in Type I problems, when most subjects likely would quickly recognize the single feature discrimination. To capture this kind of knowledge, we added a simple encoding of relevant feature detection. A feature can be considered relevant to the prediction when an ignored (i.e., abstracted) feature leads to an incorrect prediction. The SCA model notes this situation and considers the feature that was ignored a relevant feature and abstracts it last, rather than randomly, in the future. This relevant feature detection allows the model to learn Type 1 (single dimension) categories more quickly. The chosen feature can change as the model is run, so that a model attempting to learn a Type 6 category will continue to try different features as relevant.

Figure 7 illustrates the ATC+SCA model results (30 model runs per problem type). The results reflect a poor fit to the aggregate human data; the G^2 aggregate fit statistic is 674. While there is a category and block effect—duplicating the qualitative results of Miller (1993)—the SCA mean learning rate for each problem type was much faster than the mean of the human subjects.

5.3 SCA as a Model of an Individual

Given the constraints imposed by methodology and architecture, we turned to the human data to understand what humans were learning during task performance and to provide guidance in adapting SCA to fit the human results quantitatively. Other solutions might be to consider new learning algorithms; one is introduced in the next section. However, because we are embracing the constraints of an architectural approach, it was important to explore further the possibilities of explaining the results within the context of the existing architecture and category learning model.

The BBN analysis of initial results focused exclusively on aggregate data. We examined the learning trajectories of individual subjects to determine if they corresponded qualitatively with the learning trajectories of individual model runs. Figure 8 plots the human individual data for Types 1, 3 and 6 category learning

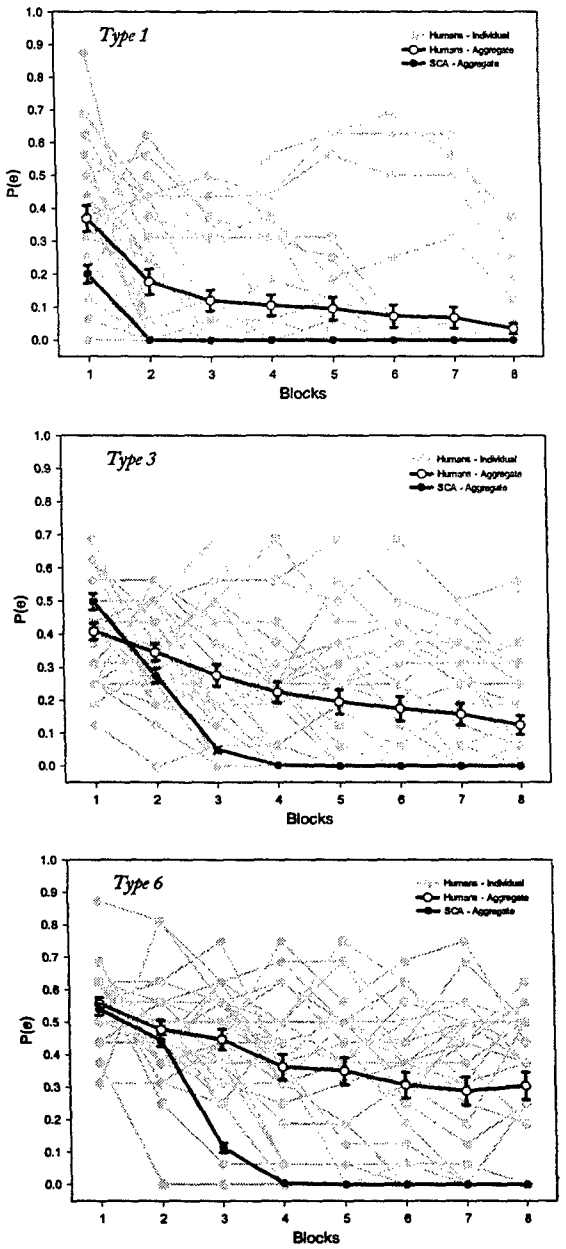


Figure 8: Individual and aggregate human data compared to the aggregate SCA data.

in the ATC task. These figures more effectively communicate the large variation in human subject learning trajectories. For example, for Type 1, four subjects failed to recognize the relevant feature by the end of the final block; the error of just these 4 subjects accounts for the error in block 8; other subjects had learned the category. For Type 6, one subject learned the correct classifications by block 2, while almost a third of the subjects were still performing at chance in the eighth block. Further, the trajectory of no individual human learner matched the shape of the aggregate human learning curve. This observation is important because it

highlights the importance of our overall philosophy. Tuning the learning to match the aggregate learning might result in improved fits, but would not likely shed further light on human behavior, because individual behavior is distinct from the aggregate in this task.

In the SCA model used to generate Figure 7, the model knowledge is identical across all model runs and there are no learning parameters other than this (constant) model knowledge. Hence, the initial SCA model is better viewed as an individual model rather than models of a population of subjects. Although SCA's learning rate appeared much too fast in comparison to the aggregate, the SCA results for the three problem types are within the bounds of the fastest and slowest human learners. Individual SCA learning curves also qualitatively matched the shape of some individual learning curves. SCA provided an exact fit to nine Type 1 subjects and to at least one subject for each type if the first block (essentially random guessing) is ignored. Thus, some SCA individual runs matched individual humans.

The limitations of creating models that match aggregate data alone are widely recognized (Estes, 2002). The analysis at the level of individual learners provided evidence that the SCA model might not be nearly as poor as one-dimensional comparisons to aggregate data suggested. This analysis provided the impetus to continue using SCA to model the ATC learning task. We now discuss improving the aggregate fit by introducing more complex feature mappings and simulating knowledge differences in subjects.

5.4 More Complex Feature Mappings

One major difficulty in modeling learning experiments is that models tend to focus only on a restricted set of features, those to which the experiment instructed the subjects to attend. However, the brain interprets its environment, notices features, and learns continuously. Subjects (consciously or unconsciously) are likely detecting, and possibly using, all sorts of features aside from the ones instructed in the experiment.

The initial SCA model for the ATC task employed only three binary-valued features to represent the feature space for the learning task. Because SCA performs a refinement search over feature space when learning, its learning rate is sensitive to the number of features; they define the size of the search space. For the ATC experimental conditions (3 features and 2 classifications), the size of search space is only 54 prediction rules. One of the reasons SCA learns so quickly is that the size of the search space is so trivial.

We empirically and mathematically explored the sensitivity of SCA to the number of binary-valued features. For example, for 4 features, the learning rate is not significantly affected and the total search space remains modest (162 total rules). For 6 features, however, in the 8th block, probability of error is only slightly better than chance for Type 6. For 6 binary features, the total search space is 1,458 total rules and, thus, after 128 trials, less than 10% of the search space will have been explored. This analysis demonstrates that the introduction of additional features would slow the learning rate dramatically, and thus potentially improve the fit to the aggregate human data.

S#	Responses suggesting a consideration of factors other than fuel, turbulence, and aircraft size
1	Other features: "...I thought it might be the direction, the area it was in, the location of nearby planes and the amount of fuel for the size of the plane..."
19	Other features: "...I also took into consideration the direction the plane was moving..."
24	Weighted factors: "...it took me several rounds to discover the importance of the turbulence rating. Before I discovered this, I paid more attention to fuel and size"
30	Constructed features: "The planes that had all the lowest descriptions together or the highest descriptions all together were accepted."
39	Semantic interpretation: "High turbulence meant to me that smaller aircraft could not change altitude. "
41	Constructed features: "I rejected all double positives and double negatives i accepted the ones only with a negative and a positive"
46	Semantic interpretation: "It would make sense that a small plane with low fuel with high turbulence would want to change altitude, and be granted that right in succession."
51	Other features: "...partially by locale."
58	Semantic interpretation: "It was hard not to think logically about whether or not the planes should be allowed to increase their altitude. Smaller planes with less fuel and a high turbulence, to me shouldn't be allowed an increase on their altitude."
60	Other features: "...how close they were to the intersection with other planes..."
64	Semantic interpretation: "I felt my strategy was more of common sense too; a small plane experiencing heavy turbulence and light on fuel would definitely need make some adjustments."
77	Other features: "...I used percent of the fuel and direction of the airplane as cues..."
83	Other features: "...First i thought you had to change the altitudes when two planes were about to crash. Then I thought it dealt with the N/S and E/W directions..."
86	Constructed features: "After the first couple of trials, I noticed a pattern. For example, I knew that if it was 20 S 3 it had to be true and if it was 40 S 3 it had to be false. I just assumed the opposite: If 20 S 3 was true, then 40 S 3 had to be false and so on."
87	Other features: "...At first my strategy was more complicated than necessary. I looked at the direction of the plane, and chose reject for each, until I discovered which was correct in each direction..."

Table 1: Human subject self-reports of their learning process.

An obvious possible source of additional features is the additional information available on the screen. Within the immediate vicinity of the instance features to which subjects were instructed to attend are the iconic representation of the aircraft (pointed in one of four different compass directions), a text string representing the airline name, and a three digit flight number. Subjects were given explicit instructions to ignore all but the instance values but some human subjects reported that their hypotheses and learning were influenced by these additional factors in the post-test questionnaire. Table 1 lists the subjects that reported being influenced by additional stimuli. Further, because participants likely perceived this information, it is still possible that it influenced their categorization processes even if not reported (e.g., some sub-

jects answered the question about strategy change with "my strategy did not change after I determined the pattern").

In addition to considering extra features, Table 1 also shows that subjects considered still other factors when performing the decision task. Some constructed new features from the combination of features (e.g., Subject 30's consideration of "all high" or "all low" inputs). Some may have considered the type of the values. For example, one subject reports aircraft with 20 gallons of fuel, rather than 20% of its fuel remaining. Many reported being influenced by the meaning of the features in the context of the task, such as Subject 58, who thought that small planes in high turbulence should be allowed to change their altitude, based on a semantic interpretation of the features. Close to 20% of the subjects in the study reported being influenced by one of more of these additional factors.

Further, the features themselves could also be considered as more complex than a simple attribute. Unlike the orthogonal stimuli used by Nosofsky, et al. (1994), all of the feature values are alphanumeric, suggesting more fine-grained discrimination between features could be necessary. Some co-occurring feature values have similar shapes (e.g., "S 3" and "L 1") and the fuel value is represented by 2 digits ("20" or "40"). Thus, it is plausible that more than just a single feature could be associated or constructed from an individual input value.

These examples illustrate that information excluded or not considered in the instructions can (and did) influence human subjects. However, the individual data subjective reports are anecdotal and insufficient for determining how to enumerate and codify these effects. To resolve this lack of specificity, we chose to simulate these effects with a number of additional, random values. These values are thus normative rather than descriptive, but are meant to capture the effect of attending to non-relevant features, considering the meaning, type, and interrelationships among relevant features, constructing new features from combinations of relevant features, and the possibility of perceptual discrimination issues among the alphanumeric feature values.

A better, more detailed model would explain how and why subjects create, consider, and attend to additional features. However, a lack of precision at this level of specification is a limitation of all models of category learning, not only SCA. By adopting this approach, we simply introduce the number of features as a model parameter. However, one of the positive consequences of the constraint of the architecture and model is that they led us to consider these issues. Because the architecture lacks other parameters that might mask these effects, SCA predicted that additional features would play a role in human learning.

5.5 Abstraction Strategies

Looking at the individual data also led us to consider a number of potential abstraction orderings, rather than the single method used initially. We observed that for Types 3 and 6, some human subjects exhibited

steady progress to zero error, some subjects made little improvements after repeated trials, and some regressed, exhibiting decreasing error for a number of blocks and then suddenly increasing. These patterns corresponded qualitatively to the three possible options for abstraction order outlined previously. A model with a fixed or systematic abstraction order will converge relatively quickly to zero error, even when critical features are abstracted early in the abstraction process, because less of the total feature space needs to be examined. A random abstraction order results in relatively slow progress because a much greater portion of the feature space will be examined. This unsystematic strategy leads to very slow progress when the number of features (and thus the feature space) is larger. Finally, for Types 3 and 6, relevant feature detection can lead to increases in the error. Even when the relevant feature is incorrect, it will stabilize abstraction order for a time and a consistent portion of the feature space will be examined, leading to a decrease in the error. However, when the model recognizes another relevant feature candidate, it changes the abstraction order and moves to a different part of the feature space. This move can increase the error because the model may have learned few prediction rules in the new area of the feature space.

5.6 Populations of Models

Given the possibilities outlined above, we now had twelve options for instantiating an SCA model (i.e., 0 to 3 extra features and 1 of 3 different strategies). These options spanned the variation in human learning and qualitatively matched specific learning trajectories. The result is a population of models for the category learning task. Because, as we described above, the human data was insufficient to provide specific guidance for choosing distributions of features and strategies, model instances were instantiated randomly from a uniform distribution of the population of models. As before, the model was run thirty times for each problem type using the uniform distribution for each category.

5.7 Model Results

The model was fit only to the learning rate by problem type data; i.e. the human data in Figure 7, and to the subject workload data (described below). All other matches to the human data are realized without any parameter fitting for the additional phenomena; they arise completely from the fit obtained against the basic human data and the model itself.

Figure 9 illustrates the extended SCA model results for Types 1, 3 and 6. These results provide excellent fits for Types 1 and 3, and a reasonable fit for Type 6. The G^2 statistic for the aggregate fit is 9.96. Qualitative fits improve as well. Although non-uniformly distributed parameter values can improve the fit further, the uniform distributions of model types and feature vectors provided a good fit to the data with minimal additional assumptions. Plots of individual model data, shown in Figure 11, reveal a similar distribution to the individual human.

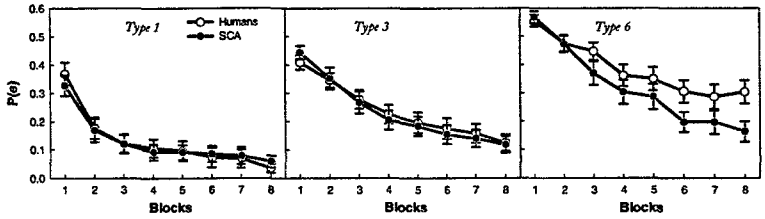


Figure 9: Extended SCA learning results (G^2=9.96).

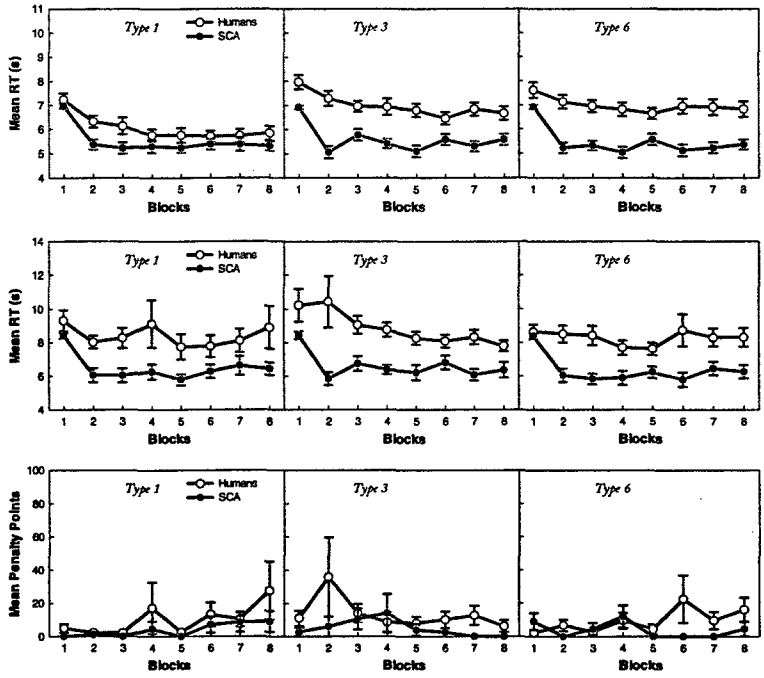


Figure 10: Some predictions of the extended SCA model: primary task mean RT (SSE=39.33); secondary task mean RT (SSE=114.73); secondary task penalty points (SSE=2720.29).

Figure 10 shows other performance predictions of the model. The top plot graphs the mean reaction time to respond to a category learning trial.[3] As would be expected, after a few blocks, in the aggregate humans respond faster to Type 1 instances than Type 3 and Type 6. However, SCA fails to capture this effect, responding in roughly the same amount of time for each type of instance. This omission occurs because the abstraction process (the loop in Figure 4) is insensitive to problem type, and thus the time to respond, within SCA itself, will always be comparable. A model that more deliberately considered strategies and options when switching to the classification task might naturally account for differences in reac-

[3] This time is measured from the moment a blip turns MAGENTA until the subject/model clicks the SEND button to complete the message. Thus, these mean RTs include the perceptual-motor time of composing the message. The time determine a prediction alone was not recorded.

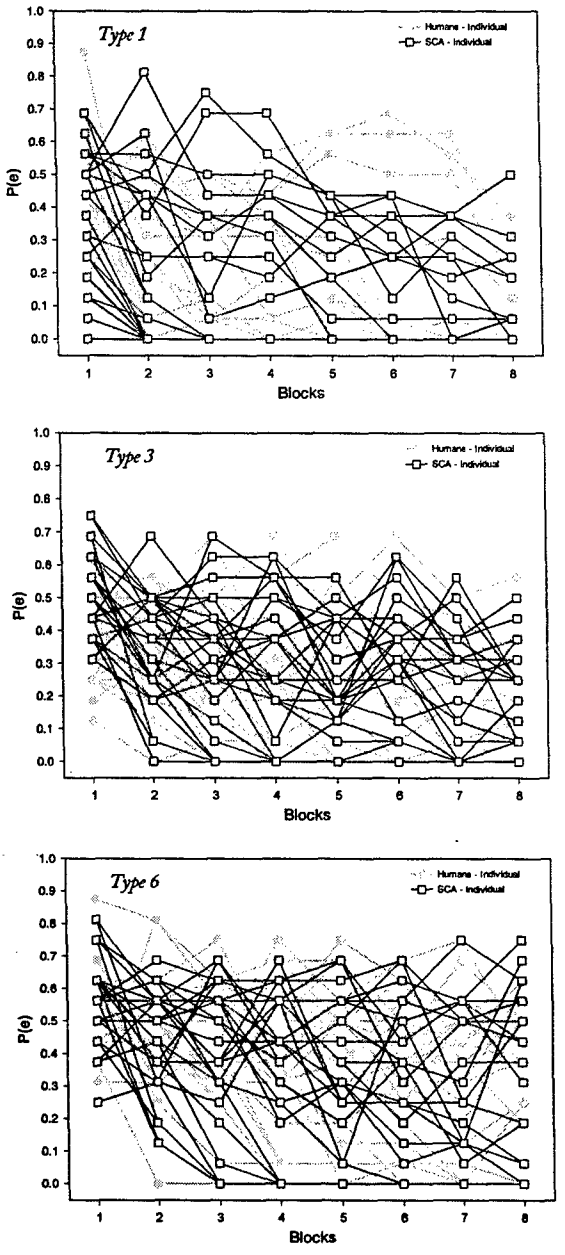


Figure 11: Comparison of human and model individual learning data.

tion times between Type 1 and Types 3 and 6, because Type 1 tasks, for most subjects, will quickly be perceived to be easier and thus require less strategic deliberation.

The middle plot in the figure is the mean reaction time measured from when a blip changes color and when the SEND button is pressed for the ATC task. The bottom plot shows the mean penalty points accrued for the ATC task.

Nosofsky, et al. (1994) showed that human subjects learn to classify Type 3 "peripheral" stimuli more slowly than "central" ones. SCA was previously shown to replicate this effect qualitatively (Miller, 1993). Figure 12 shows quantitative predictions for the AMBR learning task.

5.8 Number of Perfect Learners

Analysts at BBN introduced the notion of a *perfect learner*, a subject whose block 8 error was zero. Table 2 shows the number of human perfect learners and SCA perfect learners. Prior to its extension, all SCA model runs reached zero error by the 8th block. The number of SCA perfect learners in the extended model was nearly exact for Type 1 and Type 3 but off by a factor of 2 for Type 6. Comparing the number of perfect learners is important as a simple measure of the variability captured by the model. For example, models tuned very closely to the aggregate learning data might show many fewer perfect learners because the error in the last training block for all three problem types is greater than zero.

Problem Type	Humans	SCA, Original	SCA, Extended
1	24	30	23
3	15	30	15
6	6	30	12

Table 2: Number of perfect human and model learners by problem type.

6.0 MODEL 2: RULEX-EM

Many of the existing category learning models commit to learning as the acquisition of rules (hypotheses) or exemplars exclusively. As is the case with many dichotomies, progress often occurs with a theory that can best reconcile alternative perspectives. Empirical work by Minda & Smith (2001) shows that both representations are used in learning and identifies characteristics of the category that can cause one representation to be more prominent than another. Erickson & Kruschke (1998) and Anderson & Betz

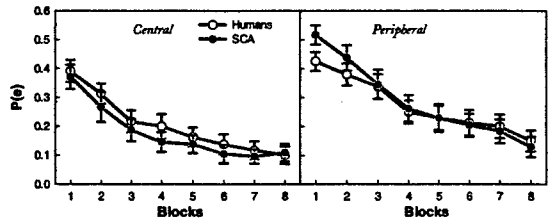


Figure 12: Prediction of "central" and "peripheral" Type 3 stimuli (G^2=3.46).

(2002) are examples of work to reconcile both representations into a coherent model. Similarly, we developed a model that employed both representations.

The hypothesis-testing process model, RULEX, was selected as a starting point because it is a process model and had demonstrated good fits to rule-based classification learning (Nosofsky, et al., 1994). The end product is a distinct process model that incorporates both rules and exemplars, includes memory effects such as forgetting, and relies on a smaller, more principled set of parameters. The model we developed is called RULEX-EM, reflecting the addition of Exemplars and Memory constraints.

6.1 Model Description

Like RULEX, the model uses a homogeneous representation for both exemplars and rules. Both are four-tuples, consisting of the three instance features (FUEL, SIZE, TURB) and an associated category (ALLOW or DENY). Both declarative representations are subject to forgetting through the base-level learning mechanism.

Exemplars are defined as fully specified four-tuples: values of all three instance features and the category (determined after receiving feedback) are specified. For example:

EXEMPLAR: [FUEL = 20; SIZE = S; TURB = 3; CATEGORY = ALLOW]

The system contains two kinds of rules. A single-feature rule is one where the value of only one of the three instance features is specified and the remaining features are unspecified (shown below as a "*"). The following single-feature rule applies to all instances where TURB is 3:

SINGLE-FEATURE RULE: [FUEL = *; SIZE = *; TURB = 3; CATEGORY = ALLOW]

The second kind of rule, an exception rule, is a two-feature rule. Following RULEX, an exception rule is a specialization of a single-feature rule; it tests a feature in addition to the one tested by a failed single-feature rule. The following exception rule could be derived from the single-feature rule above:

EXCEPTION RULE: [FUEL = *; SIZE = L; TURB = 3; CATEGORY = DENY]

A block diagram of the model is shown in Figure 13. Like RULEX, the model has two distinct phases—prediction followed by learning. Beginning after an instance appears and is perceived, the prediction phase tries prediction strategies—from specific-to-general—to determine the category of the instance. This approach, inherited from RULEX, is similar to SCA's specific-to-general search for prediction rules. Unlike RULEX, the first step is to try to recall an exemplar for the given instance. If successful, the category specified in the CATEGORY slot of the recalled exemplar is produced. Because exemplars are subject to forgetting, this recall strategy can fail.

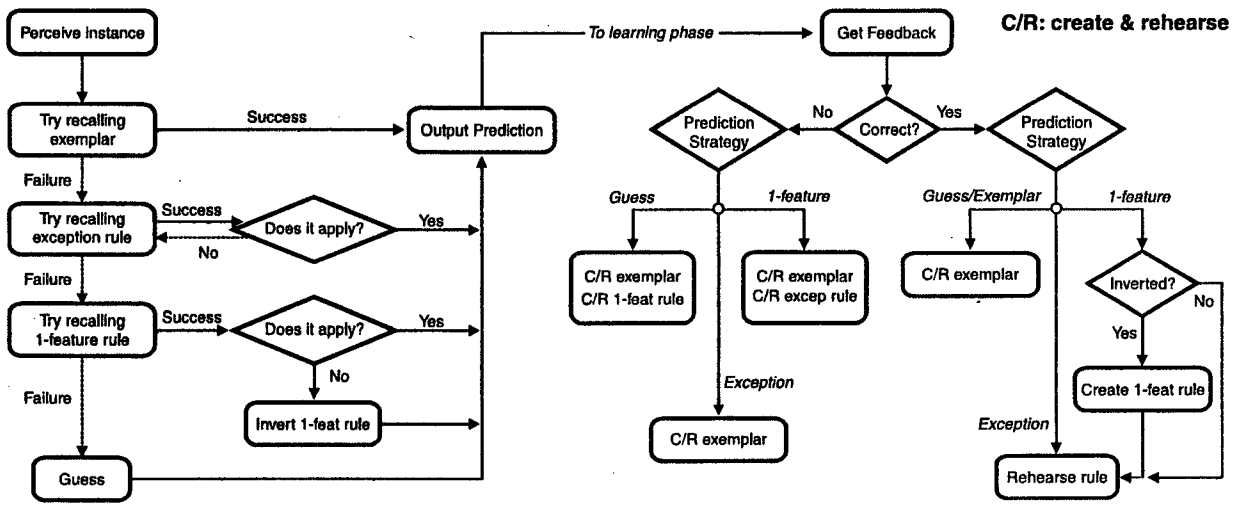


Figure 13: Block diagram of the RULEX-EM model.

If so, the model tries to recall an exception rule. If several exception rules can be recalled, the model tries the most highly activated rule. In general, this is the rule most frequently and/or recently used. If the recalled rule can be applied, a category prediction will be output.

If the exception recall strategy fails (either because no exception rules could be recalled or because none could be applied), the model then tries to recall a single-feature rule. If no single-feature rules can be recalled, the model will guess the instance's category. If a rule can be recalled and applied, the category prediction is output.

If a single-feature rule is recalled but cannot be applied, the model will produce the opposite category as specified in the rule. We surmised that subjects, realizing that the feature values and categories in this task are binary, might choose a category response that is the complement of the one specified in the recalled rule. Subject 86 in Table 1 reports performing such an inversion. To illustrate, suppose the instance is [20 S 1] and the model only has the single-feature rule shown above. The rule does not match because the TURB values are complements. A reasonable response might then be to produce the complementary category; i.e. DENY.

After a prediction has been made and feedback provided by the environment, the model enters the learning phase. The learning behavior used depends on the feedback and the strategy used to make the prediction.

If an incorrect prediction was made, the model will always create an exemplar, as defined above. (Creating a duplicate of an existing exemplar or rule results in an increase of the activation of the existing exemplar or rule.) Next, the model rehearses the exemplar. Rehearsals boost activation to increase the likelihood a memory element will not be forgotten.

If the incorrect prediction was due to:

- a *guess*, the model will create and rehearse a single-feature rule by randomly selecting one of the features of the instance and associating it with the current category.
- a *single-feature rule*, the model will create an exception rule. An exception rule is derived from the failed single-feature rule by using the specified feature value in the rule and one other randomly selected feature value from the instance. The exception rule shown above could have resulted from the presentation of [20 L 3] and the recall of a previously learned but incorrect single-feature rule.

If a correct prediction was produced, then the learning behavior is dependent on the prediction strategy, as follows:

- a *guess* or *exemplar recall:* the model creates and rehearses the exemplar.
- an *exception* rule: the model rehearses the exception rule.

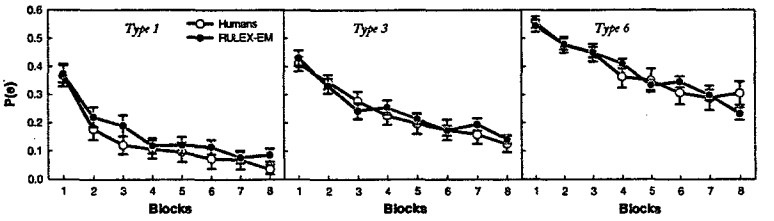


Figure 14: The fit of the RULEX-EM model to the human learning rate data. Error bars designate 95% confidence intervals. (G^2=5.64)

- a *single-feature* rule: If a single-feature rule was directly applied, the rule is rehearsed. If the model took the complement (or "inverted") a single-feature rule, then the model creates and rehearses a single-feature rule representing the inverted rule.

Due to the architectural base-level learning memory mechanism, exemplars and rules can be forgotten unless they are used or rehearsed. Rules that are predictive will be chosen and used more often, further increasing the chance of being used and not forgotten. Rules that are not predictive will experience less use and will eventually be forgotten. In contrast to rules, successful exemplar recall will always produce a correct prediction. Therefore, their activation will increase largely as a function of the number of exposures.

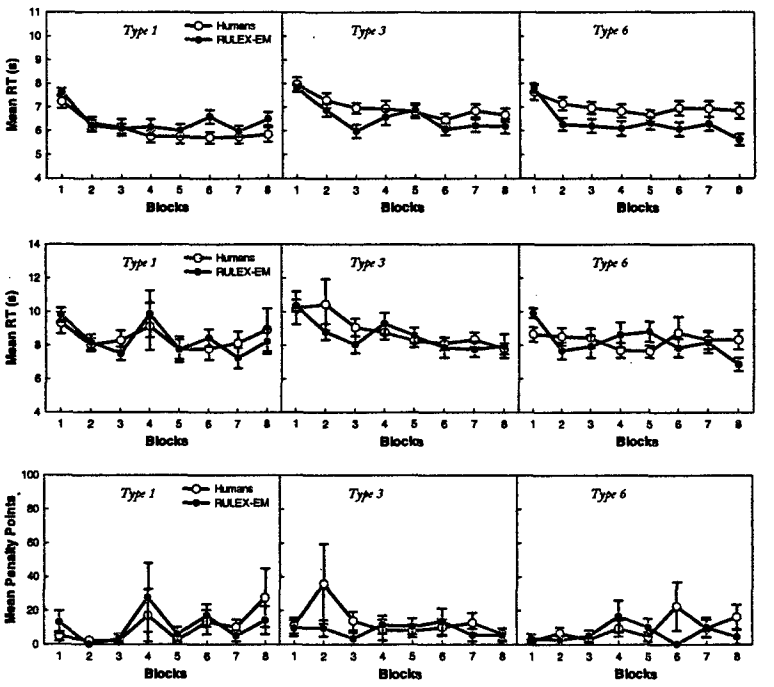


Figure 15: Some predictions of RULEX-EM: primary task RT (SSE=8.40); secondary task RT (SSE=15.24); secondary task penalty points (SSE=2043.46).

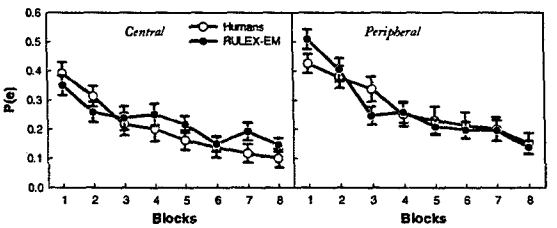


Figure 16: RULEX-EM prediction of Type 3 learning rates for "central" and "peripheral" stimuli (G^2=5.89).

6.2 Model Results

Figure 14 shows the fit of the model to the learning data as a function of problem type. Aside from the four parameters of the base-level learning mechanism (which were never manipulated), these fits are accomplished using only two free parameters: the rehearsals used to reinforce exemplar and rule memories. The best fits were achieved with rehearsals of four and seven, respectively. A goodness-of-fit analysis produced a G^2 of 5.64. The human data and model show significant effects of problem type and blocks. Figure 15 presents a series of model predictions. For the primary task reaction time (the time from when a blip turns magenta until when the "send" button is clicked), the human data showed significant effect of blocks; subjects performance is improving with practice. However, there was a weak effect by type; Type 1 is different from Types 3 and 6, while Types 3 and 6 are not different from each other. For this measure, the model was only able to reproduce the effect by blocks. There was a significant effect by block for secondary task reaction time for both humans and the model. Finally no effect by problem type or block was found for the mean penalty points in humans; there was a similar absence of effect for the model data.

Figure 16 presents the model's central vs peripheral prediction. There were significant effects of stimulus type and blocks for both the human and model data. Figure 17 compares individual human and model data. The model's variability in performance decreases as problem difficulty increases, unlike the human data or the SCA model (Figure 10).

6.3 Insights Provided by the Model

To further evaluate the model and to gain a deeper insight into its behavior, we instrumented the model to collect data on prediction strategy utilization as a function of experience and problem type. Figure 18 shows the collected data as a distribution of strategy use in the model for the three types of problems. Initially, guesses are frequent, but quickly taper off as learning progresses; guesses are more frequent for the difficult Type 6 problems. Single-feature rules are learned quickly and persist in the Type 1 condition, as expected. However, for Type 6, single-feature rules are tried, demonstrate little utility, and yield to exception rules. (Recall that exception rules are created when single-feature rules fail.) Exception rules

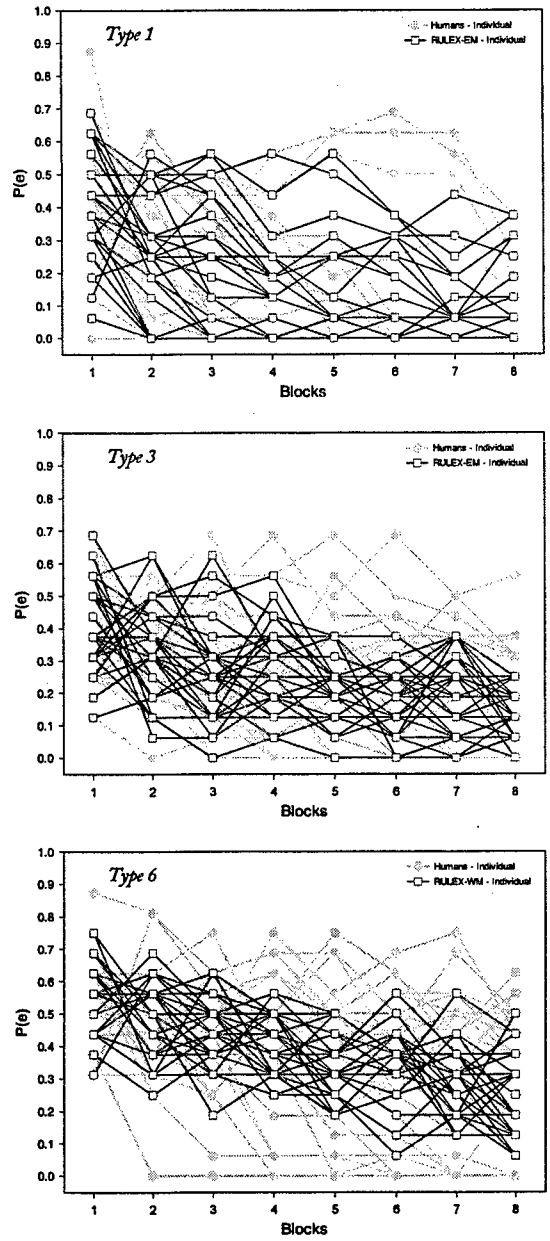


Figure 17: Comparison of human and model individual learning data.

grow to dominate by the end of learning in Types 3 and 6, as would be expected. Exemplar recall is used more in Type 6 problems than Type 1 because the difficulty of Type 6 causes more incorrect predictions, which leads to increased opportunities to memorize and rehearse exemplars. The steady increase in exemplar recall mimics implicit learning or priming effects that can occur merely from repeated exposure to stimuli.

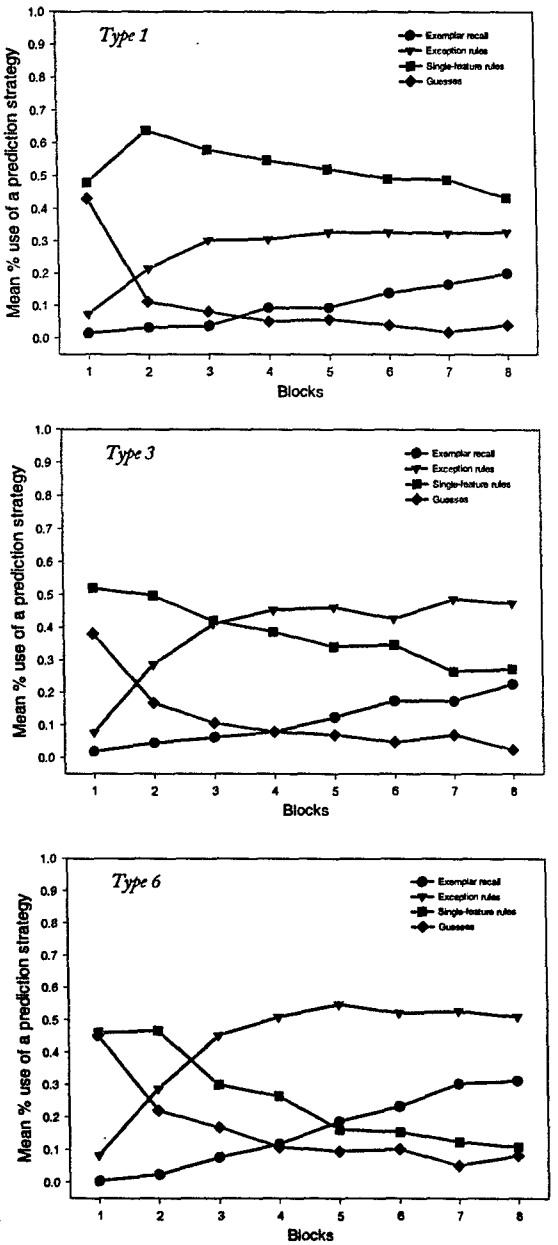


Figure 18: Evolution of strategy use across blocks by problem type.

7.0 SUBJECTIVE WORKLOAD

The AMBR models were required to report subjective workload. While there are many ways to compute subjective workload, our measure is based on the *realization* that there is "work" to be done. This formulation was also used in AMBR Phase I.

Triggers	Work to be done	Workload value
Orange	Transfer outgoing blip	3
Yellow	Tell outgoing blip to contact ATC	4
Green	Accept incoming blip	3
Cyan	Welcome incoming blip	1
Red	varied	10
Magenta	Allow/Deny speed change request	10
Negative feedback	Need to learn the category better	15

Table 3: "Work to be done" and associated workload values.

In this model, blip color signals when task actions are needed, therefore, realizations are based on blip color. Table 3 also shows the workload values associated with each kind of work. The values represent the importance or urgency of the tasks, relative to one another, as can be deduced from the task instruction and the penalty point schedule. These values are used in this workload equation:

Equation 2: $$w(t) = \alpha * \Sigma l_i / t$$

This formulation captures two intuitive characteristics of workload: it is a function of the amount of work to be done per unit time, and also of the difficulty or urgency of the work to be done per unit time.

For each realization, i, the model records the associated workload value, l_i. At the end of a model run, the total workload, Σl_i, is divided by the duration of the scenario in seconds (600sec), t, and multiplied by a scaling factor, α. The scaling factor was selected to provide the best fit to the empirical data; a value of 5. Although we prefer predictions over fitting, we knew of no pre-existing work on model-generated subjective workload work from which to borrow ideas or parameters. Instead, we developed our own model (Equation 2) and fit the results to the data. The model fit is shown in Figure 19.

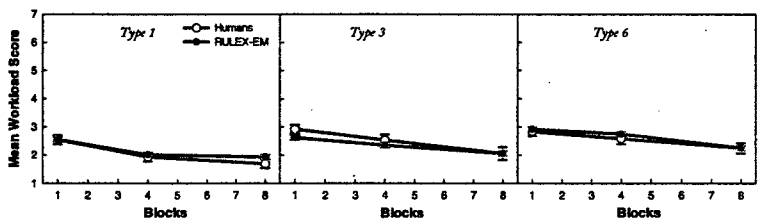


Figure 19: Workload (SSE = 0.21) for the RULEX-EM model.

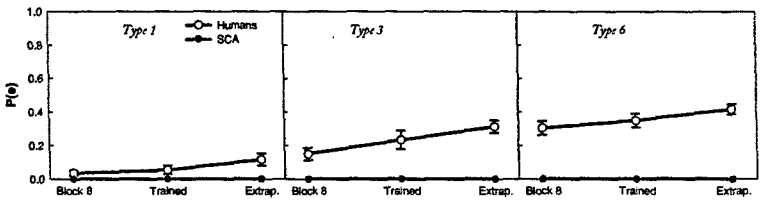


Figure 20: Initial SCA transfer task results in comparison to human results (G^2=420.09). Probability of error is plotted for the last training block ("Block 8"), the trained instances during the transfer task ("Trained"), and for those instances that could be unambiguously mapped to trained instances ("Extrap").

8.0 TRANSFER TASK

We have examined some predictions of SCA and RULEX-EM in the context of the eight learning blocks. This section reports on the model predictions for the transfer task described.

8.1 SCA Transfer Task Results

Figure 20 shows the initial transfer task results for the SCA model. For this model, knowledge was added to map unknown feature values to the values specified in the task instructions. This knowledge reflects common sense knowledge about the values of scalars and sizes. For instance, the value 10 is closer to 20 than 40; therefore 10 will be mapped to 20. It seemed plausible that subjects would use this kind of knowledge to map an extrapolated value to a known value for a prediction; thus, FUEL=20, SIZE=XS, TURBULENCE=3 would be mapped to FUEL=20, SIZE=S, TURBULENCE=3, a trained instance, and the subject would respond with the prediction learned for this instance.

There were two unresolved issues. First, mapping intermediate values (e.g., 30) included three obvious possibilities: 1) map to the lower value, 2) map to the higher value and 3) don't change the value. Thus, an instance with a fuel value of 30 could get mapped to the known values of 20 or 40 or not changed at all. For the transfer task, we designed the model to make a random choice among the three options, each option having equal probability.

Second, we considered capacity limitations on the mappings. For example, an instance of 10,4,XL requires 3 mappings (to 20,3,L) and then the retrieval of the prediction rule for this trained instance. There is no ambiguity in this mapping, but rather a question if subjects could readily perform all three mappings and then retrieve a prediction for the instance. This last issue is important because perceptual cues might be included in prediction rules (if so, subjects might respond differently to trained instances presented, verbally, for example.) Because it was the simplest approach (requiring no additional assumptions about

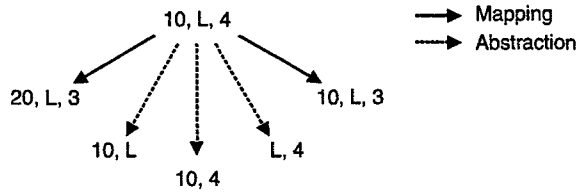


Figure 21: Example of the competition between abstraction and mapping.

capacity limitations or perceptual issues), the initial transfer task model completes all mappings and then makes a prediction.

Superficially, the SCA transfer task results appeared quite discouraging. However, the overall results are skewed by the learning rate in the original SCA model with respect to the basic learning task. Because SCA learns all categories by the eighth training block, its probability of error for the learned instances will be zero and all its transfer task predictions will be based on completely learned categorizations. A comparison of human perfect learners to the SCA transfer task results was much more encouraging. As shown in Figure 22, perfect learners produce an average probability of error close to zero for the trained instances, particularly for Types 1 and 3.

The real failure in the transfer task results is the predictions SCA made with respect to the extrapolated stimuli. SCA shows no change in probability of error for the extrapolated instances vs. the trained instances. These results are attributable to the complete mapping we chose. With the complete mapping, all extrapolated instances are mapped to the corresponding trained instances, and thus will result in the same probability of error and consistency as the trained cases. Therefore, for the revised model, we sought a simple approach that reduced the number of mappings.

In the original model, the mapping occurred before any prediction. That is, when the model was presented an instance such as (10, L, 4), it would complete the mappings to the trained instance, (20, L, 3), before attempting to make a prediction. The only change in the revised model was to allow prediction and mapping to compete. Figure 21 illustrates the process. If the model abstracted the size feature in this example ("L"), it then would consider abstracting either of the remaining values, or mapping them to training values. Prediction is accomplished though the deliberate choice to ignore (abstract) a feature, which

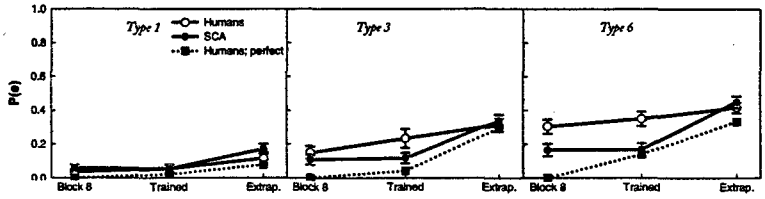


Figure 22: Final SCA transfer task probability of error results in comparison to human results (G^2=14.37).

then allows the model to retrieve any prediction rules matching the partial instance. We changed the mapping knowledge so that the model could, with any decision, choose to map features or abstract them. Abstraction operators are proposed at the beginning of the prediction process as well as any mapping operators (as before, equidistant values lead to the proposal of three mapping operators: map lower, map higher, do not map). The result is that sometimes a particular feature is mapped, and sometimes it is ignored.

Figure 22 displays the final SCA probability of error results for the transfer task. The result is a much better match to the extrapolated stimuli than was observed in the first round. However, the model fails to predict the increase in error in the trained stimuli when presented during the transfer task. SCA provides no inherent explanation of this effect. SCA predicts that the error rates should be the same across the block 8 and "trained" stimuli in the transfer task. There are a number of potential explanations for the increase in error. The methodology in the transfer tasks is slightly different and some subjects took a break before completing the transfer task, which would have led to increased delays for these presentations. An intriguing possibility is that subjects are learning something during the transfer task that interferes or inhibits their ability to retrieve their correct predictions. If this hypothesis was true, then performance should degrade over the course of the transfer task. Yet another possibility is that some subjects guessed wildly during the transfer task and skewed the results of those subjects who took this task more seriously. Answering these questions might provide some guidance towards extending the model to account for this result. However, SCA alone would not account for these differences and its failure to account for this trend in the data represents incompleteness in its account of category learning.

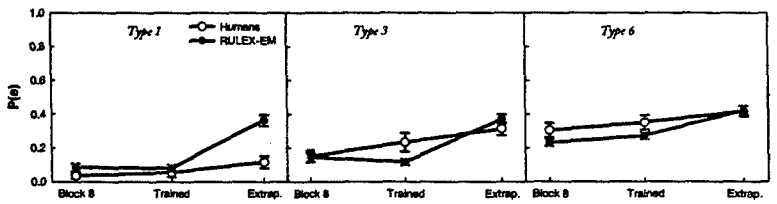


Figure 23: RULEX-EM transfer task results in comparison to human results (G^2=16.23).

8.2 RULEX-EM Transfer Task Results

We used the same transfer task implementation as used in the extended SCA model, the procedure illustrated in Figure 21. Figure 23 presents the results. First, the predictions for extrapolated stimuli ("Extrap.") are puzzling because RULEX-EM used the same mapping and abstraction process as for the extended SCA model. We presently have no explanation for its nearly uniform prediction across problem types. There might be an unanticipated interaction in the mapping/abstraction procedure and the memory

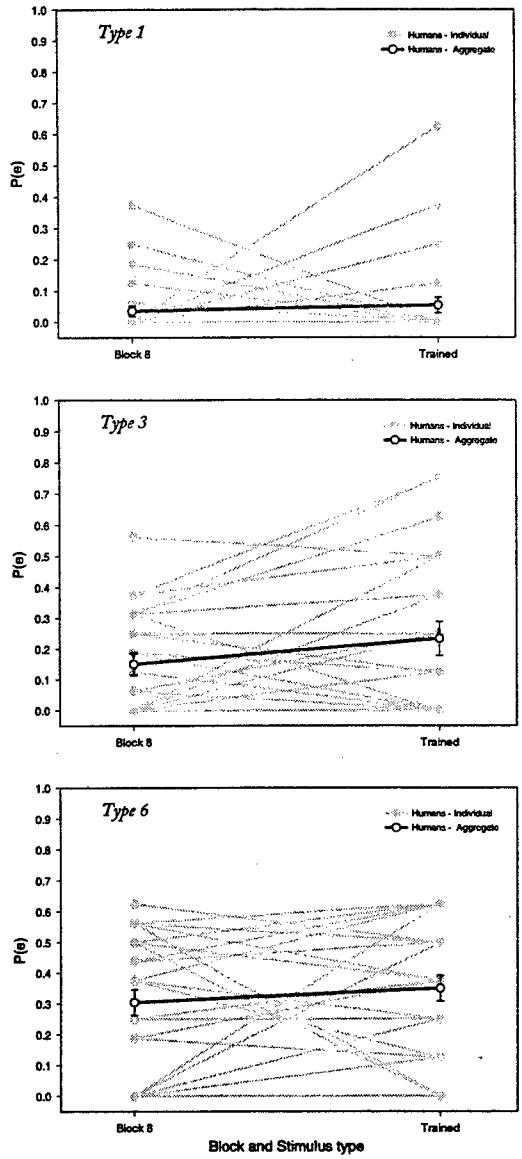


Figure 24: Individual and aggregate human data for the transfer task.

effects present in the RULEX-EM model (and not present in the SCA models). This warrants further investigation. The remainder of this section will focus on the prediction of "Trained" instances during the transfer task.

For Type 6, The model correlates well to the human data for Type 6. However, for Types 1 and 3, the model predicts small *improvements* in performance "Block 8" and "Trained" whereas the data reveals a decrement in performance.

Although the Types 1 and 3 model trends contradict the data, there are good reasons to believe the model:

- *During the training phase*, it is possible to learn a reliable rule or successfully memorize more/all exemplars in the waning moments of "Block 8". The consequence is that performance for "Trained" stimuli will be improved relative to performance in the "Block 8".
- Types 1 and 3 (as illustrated in Figure 18) rely heavily on single feature and exceptions rules. This combination provides a low degree of specificity.[4] Rules, by definition, apply to multiple instances (e.g. a single-feature rule covers four instance). *During the transfer task*, the *use* of a rule increases the rule's activation, making it more likely to be available for the other transfer task instances covered by the rule. In essence, the model continues to learn (in the absence of feedback), resulting in improved category prediction performance for the trained instances. In contrast, Type 6 relies heavily on exceptions and exemplars. The high combined degree of specificity of exemplars and exceptions—five: three for exemplars plus two for exceptions—allows less reuse and therefore a lower likelihood of improved performance.

To gain more insight into performance on the "Trained" stimuli, we turned to the individual human data. Figure 24 shows the individual and aggregate human data. Similar to the learning data (Figure 8), there was much more variability than expected. Closer examination of the data revealed other unexpected findings. For example, the Type 1 subject with the worst "Trained" performance (0.625) had previously demonstrated perfect performance—P(e)=0.0—from block two through block eight. Also, six Type 1 subjects showed a decrease in performance on "Trained" stimuli relative to "Block 8" although they attained perfect performance no later than block five. This behavior suggests that some subjects may have been confused about what to do during the transfer task. (Recall that subjects were given no practice transfer trials, nor were they told ahead of time that a transfer task would be performed.) This confusion may have been compounded by interference between trained and extrapolated stimuli. When the transfer task data for these "outlier" human subjects are removed[5] the model's predictions of "Trained" performance are a better match to the human data, particularly for Types 3 and 6. This is illustrated in Figure 25.

[4] A single-feature rule specifies only one feature while an exception rule specifies two features, yielding a combined "degree of specificity" of three.

[5] For this analysis, we remove subjects whose performance in block eight was better than chance—P(e) < 0.5—but whose performance on "Trained" stimuli was equal to or worse than chance—P(e) >= 0.5. One subject was removed for Type 1; five subjects were removed for both Types 3 and 6.

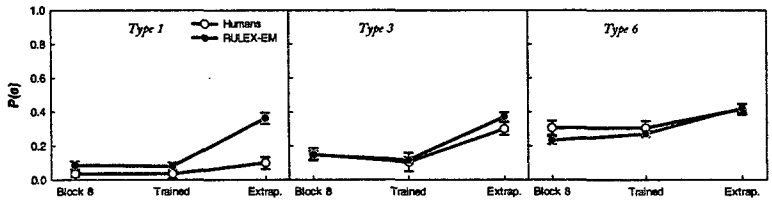


Figure 25: Revised fit after removal of outlier humans data (G^2=14.94).

9.0 DISCUSSION

"Every model is wrong, but some models are useful" is a epigram attributed to George E.P. Box. Having presented the models and some initial discussion of their fits to human data, this section examines the limitations of the models more critically, and also attempts to show, that although the models had limitations, they provided insights into human performance that made them quite useful.

9.1 Critical Analysis of SCA

The original SCA model, representing a single strategy and subject, provided learning within the brackets of the fastest and slowest human learners and matched the learning of some individual subjects qualitatively and quantitatively. The SCA Soar model provided these fits *a priori*. We achieved these results by reusing an extant model, following constraints imposed by theory, and without introducing additional knowledge or parameters. This methodology provides for the cumulation of results necessary for a comprehensive architectural theory (Newell, 1990).

We achieved improved aggregate fits by developing a population of models with different strategies (reflected in the different methods for determining abstraction order) and different feature vectors. These choices were motivated from a more fine-grained analysis of the individual data. This approach begins to approximate the demand for more sophisticated models of human learning that match not only the aggregate data but also match the learning trajectories of individual subjects and successfully predict performance on transfer stimuli (Estes 2002). The number of direct correspondences between human individuals and individual model runs does improve over the original SCA model.

One positive consequence of taking seriously the constraints of theory, architecture and a previously-existing model is that this constraint led us to consider different strategies and alternate feature encodings. While we would much prefer to be able to make a priori predictions of the features based on the representation, post hoc estimation of the feature space is consistent with the decisions of other modelers. For example, in an icon search task model, Fleetwood & Byrne (2002) indicate that the features for their task, which include very simple shapes as well as more complex icons, were estimated from human data. Currently, models of perception such as EPIC and ACT-R/PM do not inform or constrain the number of fea-

tures needed for any particular percept (in part because such results are not available in the human factors or cognitive science literature). Thus, while our approach did require post hoc analysis to match the data, it also led us to ask other questions that modelers using other architectures were not led to ask. In general, this is one of the advantages of alternate theories and computational architectures: different theories and architectures lead to different ways of interpreting and analyzing a data set, thus offering, in sum, a greater perspective on the human phenomena than any individual perspective.

While the aggregate and individual model fits were good, issues and questions related to the plausibility and completeness of SCA as a descriptive model of category learning were not resolved. If the debrief results can be assumed to be reasonable indicators of a person's self-awareness of cognition, it appears that human subjects learned using both exemplar- like techniques (Subject 2: "I started to notice a pattern") and hypothesis testing techniques that led to rules (Subject 18: "Accept if the plane's fuel was at 20, reject at 40"). SCA models only the former. A potential line of future research would be to combine the hypothesis-testing component of the RULEX-EM model with SCA and compare the results to other hybrid models.

SCA's abstraction process is deliberate, which makes the use of additional, irrelevant features more problematic for a descriptive model. Even if irrelevant features were perceived, SCA, as currently conceived, would ignore them due to their irrelevance (as defined by the task instructions). While the immediate goal was to model this task within the existing model of SCA, we have begun to investigate an alternative formulation of SCA that will use episodic indexing (Altmann and John 1999). In this model, feature abstraction occurs as a consequence of attention and recall, rather than via a deliberate abstraction process. Preliminary results suggest this model will provide similar learning results but avoid the use of a deliberate abstraction procedure, a psychologically unrealistic component of SCA.

The current model also ignores that any extra features considered in early blocks would be decreasingly likely to play a factor in later blocks. Although human subjects did report being influenced by external factors, most acknowledged or implied that they abandoned or excluded these factors as experience with the task deepened. In the attention-based SCA model, an obvious way to model the decreasing influence of extra features would be to learn to attend only to the features that led to positive feedback. Initially, this learning would introduce new features into consideration (because, by random chance, some of them would be useful for a few trials), but, over time, would allow the model to converge on just those features that are needed for categorization. Such a model will be a step towards a descriptive model that accounts for the consideration of extra features in human learning. However, such a model still would not address other factors like feature value discrimination and semantic interpretation of the features and values.

Hypotheses other than extra features will also be useful for developing deep explanations of human learning in this task. For example, in Round 1, we evaluated how learning slowed in SCA if the model did

not learn on every trial (SCA learns a new prediction rule with every learning trial). Introducing a probability of learning parameter did allow SCA to better match the aggregate learning curves. The primary reservation with this approach is that we did not find any direct evidence in the data that would explain why the model should not learn on every trial. Subjects did report that they grew bored during the experiment and for a period responded randomly or with the same response. These responses provide a clue as to how to slow learning, but again, the challenge is to capture and to encode the conditions under which such factors should be considered, requiring models of motivation and interest.

9.2 Critical Analysis of RULEX-EM

The development of RULEX-EM relied on synthesis and integration on several dimensions. First, the model is constructed within the constraints of an integrated cognitive architecture. Second, the model contains both exemplar and rule representations. Third, the learning model is integrated with a pre-existing model of a dynamic perceptual-motor performance task.

RULEX-EM, evaluated in the context of the AMBR learning and performance task, produced very good fits and many predictions confirmed by human data. We attribute the model's success, in part, to a) the architectural integration of elements of ACT-R, Soar, and EPIC, b) inheriting the validation of these systems, and c) accepting the modeling constraints of these mechanisms. Notwithstanding these positive results, there remain many unresolved limitations and methodological issues.

One such issue concerns the uniform representation of rules and exemplars as declarative four-tuples. This representation was similar to that used by Nosofsky, Palmeri & McKinley (1994) and was sufficient for modeling the AMBR learning task. However, we are not asserting that this representation is used by humans. A review of the empirical literature will help to inform the proper representations for rules and exemplars.

Another issue concerns the strict specific-to-general approach to category prediction. It is unlikely that humans use such a strict process to determine an instance's category. A consequence of the model always considering exception rules before single-feature rules is seen in Figure 18 by the very high use (30%) of exception rules in Type 1 problems, where one might expect near exclusive use of single-feature rules. We are considering an alternative prediction process; one that is based on competition between prediction strategies (exemplar recall, exception rules, single-feature rules). A competitive prediction approach—selecting a prediction strategy based on their utility (activation) and tending to favor new rules or rules successfully used in the previous trial—may produce a more believable strategy distribution. Such an approach would require a control scheme that deliberately chose not to use a recalled rule that had just failed. This would allow the model to better explore the space of hypotheses, while not preventing the reuse of previously failed rules.

A weakness of the model is the episodic recall strategy, if it succeeds, will always return the correct category. This occurs, in part, because exemplar memories are a four-tuple, tightly coupling the instance features with the category. A more ecologically correct representation should perhaps group instance features in a three-tuple, create a distinct memory structure for the category (because the feedback is temporally displaced from the onset of the instance features), then use an associating memory structure to bind the instance to the category. This dissociation of stimuli feature and category allows the model to recall or recognize a stimulus, while not ensuring the availability of the category because the associating memory structure may become unavailable due to inadequate use. While this solution provides a more distributed representation, it only addresses failure to recall a category. Another issue entirely ignored by the present architecture and model is successful, but confused recall or recognition due to the interference between similar stimuli and category classes. This is an important avenue for future architectural explorations.

The individual model data (Figure 17) is almost uniformly distributed around the mean model data. This occurs because the model's primary source of variability is noise. This reliance on noise is responsible for the unexpected reduction in variability as problem difficulty increased. Wray & Laird (2003) caution against attempts to represent the range of human behavior by noise alone. Rather there must also be representations of the variety of strategies humans perform. RULEX-EM only implements a normative strategy and it does not include specific prediction or learning strategies, such as used by a subject who may a) choose to give the same prediction until they have observed a pattern or b) makes an *a priori* decision to memorize the exemplars; e.g. one subject reported an *a priori* decision to memorize instances. It also does not include the behavior of subjects who did not conform to task instructions and considered non-relevant task features when formulating their categories (as exemplified in Table 1) or who perhaps did not understand the goal of the task (e.g., as discussed in Section 8.2.)

RULEX-EM is principally a learn-on-failure model. Only exemplar learning and reinforcement, though the creation and rehearsal of exemplars, will occur until a prediction (using the "guess" prediction strategy) fails. This learning approach has implications for modeling human data, most notably for Type 1 problems. The individual human data graphs in Figure 17 show that at least one human subject produced perfect prediction (P(e) = 0) in Block 1. In fact, this subject made only one error (in block 8) for their entire training. RULEX-EM cannot reproduce this behavior and offers the implausible explanation that this subject guessed correctly over seven blocks. In contrast, SCA, which learns on every instance presentation, is able to produce perfect prediction in Block 1 for Type 1, as illustrated in Figure 10. It may be possible for RULEX-EM to also produce this behavior with the addition of explicit cognitive strategies such as those mentioned above for increasing the variability of the model.

RULEX-EM required tuning of only two free parameters: the number of rehearsals for rules and exemplars. These two stand in contrast to the ten free parameters available in RULEX. However, before this reduction can be considered an achievement, RULEX-EM must demonstrate coverage of the breadth of data previously fit by RULEX. Additionally, the explicit rehearsals controlled by those two free parameters are an example of the "placeholder" nature of free parameters. We do not believe that subjects are deliberately and consistently rehearsing rules seven times and exemplars four times. Instead, it is more likely that subjects are performing productive processing of rules and exemplars that has an effect approximated by explicit rehearsals. As we refine the model, we hope to identify parameter-free processes that eliminate the need for explicit rehearsals.

10.0 CONCLUSIONS

Modeling, in all its forms, is a technique by which we operationalize, test, and expand our understanding of phenomena or behavior. Model development has a large number of degrees of freedom. Constraints on the process reduce the space of model development options, resulting in more principled models. One of the themes of this report has been the essential role that constraints played in the development of our models.

The methodology and principles Newell (1990) outlined for unified theories of cognition strongly constrained our modeling decision. The EASE architecture, developed in the course of the AMBR project, came about as a result of "listening to the architecture" and identifying areas where the set of mechanisms of the original architecture (Soar) needed to be amended. Also, we have reused the SCA model, developed in previous work on category learning. Although the initial fit of the SCA model to the mean human data was poor, the constraint of "listening to the architecture" and seeking solutions within it, led to further, deeper analysis of the human data. By minimizing changes to both the architecture and model, we developed the hypothesis that other factors were playing a role in the task (such as additional features and semantic interpretation of the features). This hypothesis was confirmed in the more fine-grained data analysis. The normative effects of these features were then incorporated into the existing model with little change to the model itself. By following the task and methodological constraints, we preserved the previous validation of SCA and gained a deeper understanding of influences on learning for the AMBR task.

The mechanisms of an architecture also provide strong constraints on the formulation of the model. For example, the visual perception system of EASE (inherited from EPIC) represents retinal zones and the varying availability of perceptual features and objects depending on which zone the object is located. This constraint required that the model include a saccade generation component. EASE also includes the base-level learning memory mechanism from ACT-R. When this mechanism was first added to the model, it

immediately required the addition of knowledge for coping with forgotten items. Both of these mechanisms required a more detailed representation of the task. However, the consequence is a deeper understanding of the behavior and the emergence of unanticipated effects; e.g. performance errors.

Another source of constraint present in this work the adoption of pre-existing mechanisms that have been validated in other architectures and models. EASE contains the sensory, perceptual, and motor mechanisms of EPIC, and the ACT-R base-level learning memory mechanism. In both cases, the established free parameters values were also adopted and were not tuned to fit human data. By adopting mechanisms from other architectures, EASE inherits the validation of these mechanisms. Additionally, the history of findings that validate those mechanisms and particularly the established free parameter setting acts as a strong constraint against tuning the parameters to provide better fits.

Thus far we have only addressed the architectural constraints. However, the architecture only captures the *invariant* aspects of the human organism—the mechanisms, structures, and processes that are brought to bear in facilitating all behaviors. The architecture does not produce behavior. Rather, it is *knowledge* that determines what and how behaviors are generated (Newell, 1990). Differences in knowledge and how that knowledge is applied produces a wide range of learning and performance in human behavior.

For many tasks, the greatest constraint arises from the explicit and implicit knowledge available in the task environment and task instructions. Encoding instructions and dependencies identified from a functional analysis can define much of the process necessary to perform the task. This is often true of interactive tasks or sub-tasks that have a minimal cognitive component, such as the immediate behavior tasks commonly used in psychological experiments. The model of the ATC task (Experiment 1) is an example of such a task. The explicit knowledge was encoded to produce the interactive behavior and implicit knowledge in the penalty scores informed how the model would resolve multi-task demands.

However, even in Experiment 1, covert cognitive behavior appeared to be critical to performance; e.g. the choice of a blip on which to fixate. Covert cognitive processes are also central to the learning task of Experiment 2. Tasks or sub-tasks with such significant cognitive components can seldom benefit from constraints as strong as those available for interactive tasks or sub-tasks. While a model can derive weak constraints from task analyses or related research, it is the *combination* of these weak constraints with those provided by the architecture that can result in more principled models. As architecture-based theories evolve, they will impose increasingly stronger constraints on models of cognitive behavior.

Although we were subject to many sources of constraint on the model building process, many degrees of freedom remain. This is evident by the two different models of category learning that comparably fit and predict the data. The differences representing alternate learning strategies; strategies that humans may also use. These strategies emerged from our investigation of contrasting hypotheses of the factors that influence

category learning. The SCA model investigated the effect of considering extraneous information during learning, while RULEX-EM investigated the interaction of memory effects and prediction strategy on learning. In the final analysis, both models have contributions to make to one another, with the end result being a fuller understanding of category learning.

11.0 ACKNOWLEDGEMENTS

We thank the program sponsors, other members of the AMBR program, and David E. Kieras, Randolph Jones, Anthony Hornof, Christian Lebiere, and Michael Schoelles for assistance, feedback and suggestions. Portions of this work were previously presented at the 2000 Human Factors and Ergonomics Conference, the 2002 Cognitive Science Conference, the 2003 International Conference on Cognitive Modeling, and the 2003 conference of the European Cognitive Science Society.

12.0 REFERENCES

Altmann, E. M. and John, B. E. (1999). Episodic Indexing: A model of memory for attention events. *Cognitive Science* 23(2): 117-156.

Anderson, J. R. & Lebiere, C. (1998). *Atomic components of thought.* Hillsdale, NJ: Lawrence Erlbaum.

Anderson, J. R., & Betz, J. (2002). A hybrid model of categorization. *Psychonomic Bulletin and Review,* 8, 629-647.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Chong, R. S. (2003). The addition of an activation and decay mechanism to the Soar architecture. In the *Proceedings of the Fifth International Conference on Cognitive Modeling,* Bamberg, Germany.

Chong, R. S. & Wray, R. E. (2003). RULEX-EM: Incorporating Exemplars and Memory Effects in a Hypothesis-Testing Model of Category Learning. In the *Proceedings of the First European Cognitive Science Conference*, Osnabrueck, Germany. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chong, R. S. (2000). Modeling with perceptual and memory constraints: An EPIC-Soar model of a simplified enroute air traffic control task. Final report for USAF/AFRL contract F33615-99-C-6005.

Chong, R. S. & Laird, J. E. (1997). Identifying dual-task executive process knowledge using EPIC-Soar. In the *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Erlbaum.

Erickson, M. A. & Kruschke, J. K. (1998). Rules and exemplars in category learning. *Journal of Experimental Psychology: General.* 127, 107-140.

Estes, W. K. (2002). Traps in the route to models of memory and decision. *Psychonomic Bulletin and*

Review, 9(1): 3-25.

Fleetwood, M. D., & Byrne, M. D. (2002). Modeling icon search in ATC-R. *Cognitive Systems Research,* 3, 25-33.

Kieras, D.E. & Meyer, D.E. (1997). An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction,* 12, 391-438.

Lewis, R.L., Huffman, S.B., John, B.E., Laird, J.E, Lehman, J. F., Newell, A., Rosenbloom, P. S., Simon, T. & Tessler, S. G. (1990). Soar as a unified theory of cognition: Spring 1990. In the *Proceedings of the 12th Annual Conference of the Cognitive Science Society.*

Love, B. C. & Medin, D. L. (1998). SUSTAIN: A model of human category learning. In the *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98),* pp. 671-676.

Meyer, D. E. & Kieras, D. E. (1997a). A Computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychology Review,* 104 (1), pp 3-65.

Meyer, D. E. & Kieras, D. E. (1997b). A Computational theory of executive cognitive processes and multiple-task performance: Part 2. Accounts of psychological refractory-period phenomena. *Psychology Review,* 104 (4), pp 749-791.

Miller, C. S. (1993). *Modeling Concept Acquisition in the Context of a Unified Theory of Cognition.* EECS. Ann Arbor, University of Michigan.

Miller, C.S. & Laird, J.E.(1996). Accounting for graded performance within a discrete search framework. *Cognitive Science,* 20(4), 499-537.

Minda, J. P., & Smith, J. D. (2001). Prototypes in category learning: The effects of category size, category structure, and stimulus complexity. *Journal of Experimental Psychology: Learning, Memory, and Cognition,* (27), 775-799.

Newell, A. (1990). *Unified theories of cognition.* Cambridge, MA: Harvard University Press.

Nosofsky, R. M., Gluck, M. A., Palmeri, T. J., McKinley, S. C., & Glauthier, P. T. (1994). Comparing models of rule-based classification learning: A replication and extension of Shepard, Hovland, and Jenkins (1961). *Memory & Cognition,* (22), 352-369.

Nosofsky, R. M., Palmeri, T. J, McKinley, S. C. (1994). Rules-plus-exception model of classification learning. *Psychological Review,* 101, 53-79.

Pew, R. W. & Mavor, A. S., (Eds). (1998). *Modeling human and organizational behavior: Application to military simulations.* Washington, D.C.: National Academy Press.

Rosenbloom, P.S., Laird, J.E., & Newell, A. (1993). *The Soar Papers.* Cambridge, MA: The MIT Press.

Shepard. R. N.. Hovland. C. I.. & Jenkins. H. M. (1961). Learning and memorization of classifications. Psychological Monographs, 75, (13, Whole No. 517).

Wray, R. E. & Chong, R. S. (2003). Quantitative explorations of category learning with symbolic concept acquisition. In the *Proceedings of the Fifth International Conference on Cognitive Modeling*, Bamberg, Germany.

Wray, R. E. & Laird, J.E. (2003). Variability in Human Behavior Modeling for Military Simulations. In the *Proceedings of the 2003 Conference on Behavioral Representation in Modeling and Simulation (BRIMS)*, Phoenix, Arizona.

Young, R. M. & Lewis, R. L. (1999) The Soar cognitive architecture and human working memory. In A. Miyake & P. Shah (Eds), *Models of Working Memory: Mechanisms of Active Maintenance and Executive Control*, 224-256. Cambridge University Press.